The Pennsylvania State University

The Graduate School

College of Engineering

ON THE ROBUSTNESS, EFFICIENCY AND INFORMATION FRESHNESS IN NETWORKED SYSTEMS

A Dissertation in

Electrical Engineering

by

Boyu Wang

Submitted in Partial Fulfillment

of the Requirements

for the Degree of

Doctor of Philosophy

December 2019

The dissertation of Boyu Wang was reviewed and approved* by the following:

Jing Yang

Assistant Professor of Electrical Engineering

Dissertation Advisor, Chair of Committee

Aylin Yener

Distinguished Professor of Electrical Engineering

Nilanjan Ray Chaudhuri

Assistant Professor of Electrical Engineering

Hui Yang

Associate Professor of Industrial and Manufacturing Engineering

Kultegin Aydin

Department Head of Electrical Engineering

*Signatures are on file in the Graduate School.

Abstract

The integration of physical network systems with computational capabilities has led to the new generation of systems, the cyber-physical systems (CPS) [1]. Applications of CPS range from manufacturing facilities, smart grid to robotics, smart transportation systems, etc. In contrast to traditional engineered systems, CPS emphasize the interaction between cyber space and physical environment, leading to significant improvement in system reliability, efficiency, and controllability. However, besides the benefits that CPS bring to us, the operation of large-scale CPS also poses severe challenges, among which robustness, efficiency and information freshness are three fundamental ones.

Many CPS applications are safety-critical, e.g., smart power grid, which need to operate reliably and avoid abnormal working situations [2]. Comparing to traditional engineered systems, CPS are more vulnerable to failures, faults and attacks, considering the data management and communication layer. For example, the fault in one of the networks can cause the failures of the dependent nodes in the other network, and further lead to a cascade of faults that will bring a catastrophic impact, examples include the 2003 blackout in the northeastern United States [3]. How to guarantee the safety and robustness of CPS is of critical importance.

Efficiency is another critical concern in CPS. Many CPS operates under given physical resources, and resource efficient operation is of great importance for the sustainability of CPS. For example, an electric vehicle (EV) charging network is a typical CPS, which includes a charging station that connected to the power grid and a large number of EVs. The charging station collects information and controls the charging procedure, where efficient charging scheduling is needed to reduce the electricity costs, to prevent the system from overloading, and to satisfy customers' demands.

Besides, data freshness has been playing a critical role for real-time system status monitoring and control in CPS. However, the unprecedented high dimensionality and generation rate of the sensing and control data also impose severe challenges on its timely delivery. Therefore, how to maintain data freshness at the central controller under given resource constraints in the system has become another critical yet challenging problem.

The goal of this study is to address these challenges in specific CPS, as outlined as follows.

In Chapter 1, we consider the line outage detection and identification problem in power networks. We formulate this problem as a sparse binary-valued vector estimation problem, and leverage the cluster structure existing in most multiple line outages to solve it. We propose three low-complexity graph-based algorithms to identify clustered line outages. Simulated tests in IEEE-118 bus system and IEEE-2383 bus system confirm that the proposed algorithms can significantly improve the accuracy and efficiency of baseline algorithms that do not leverage the cluster structure of multiple line outages.

In Chapter 2, we follow the problem in Chapter 1, but consider no prior knowledge of the power systems or model parameter. A pure data-driven approach is proposed to detect the events using only the frequency measurements from the power systems. Due to the complex and unknown statistic of the frequency measurements, the popular change point detection framework cannot work directly. To overcome this issue, we first use a recurrent neural network based model to predict the future measurements using the observed measurements, thus the prediction error can be monitored on-line. We note the prediction error follows to a normal distribution and the parameters can be learned by the predictor, thus makes this detection problem trackable. We assume the post change prediction error follows to a normal distribution with unknown parameters, and then this change point detection problem can be solved by many exists procedures. In this work, a Rao test based CUSUM procedure is used to recursively calculate the CUSUM statistic without complex maximum likelihood estimation calculation. We evaluate the performances through real PMU events data.

In Chapter 3, we develop an admission control and scheduling mechanism to jointly consider the revenue of charging stations and the service requirements of customers in large-scale smart grid systems. We consider an offline setting where given a set of inflexible service demands from customers. Our objective is to design an admission control and scheduling scheme, so that the admitted demands can be satisfied while the revenue of the charging station is maximized. We first propose a calculus based scheduling algorithm, and show that if the system is underloaded, it maximizes the revenue of the charging station. We then consider the general overloaded case, and prove that it is NP-complete. A heuristic algorithm is

developed to greedily decline a subset of demands until the remaining demands can be satisfied. An on-line algorithm is also proposed when the demands are unknown until them starts. We evaluate the performances of the proposed algorithms through simulations and real data set.

In Chapter 4, we consider a scenario where a source continuously monitors an object and sends time-stamped status updates to a destination through a rate-limited link. In order to measure the "freshness" of the status information available at the destination, we adopt the metric called Age of Information (AoI). We first assume all updates are of the same size, and arrive randomly at the source according to a Bernoulli process. Due to the link capacity constraint, it takes d ($d \geq 2$) time slots for the source to complete the transmission of an update. Therefore, when a new update arrives at the source during the transmission of another update, the source needs to decide whether to skip the new arrival or to switch to it, in order to minimize the expected average AoI at the destination. We prove that within a broadly defined class of online policies, the optimal policy should be a renewal policy, and has a sequential switching property. We then show that the optimal decision of the source in any time slot has a multiple-threshold structure, and only depends on the age of the update being transmitted and the AoI in the system. We then consider the various update sizes, and similar results are derived. The thresholds are then numerically identified by formulating the problem as a Markov Decision Process (MDP).

Table of Contents

List of Figures

List of Tables

Acknowledgments

No doctoral research is done alone, so here I truly wish to offer my thanks to all those who have been a part of this precious journey and would like to acknowledge a few of those people here.

First and foremost, I would like to express my deepest gratitude to my advisor Professor Jing Yang for the continuous support of my Ph.D. study and research. The two most important days of my college life is when I applied for my Ph.D. degree and when I discovered the true reason why I want a Ph.D. degree. Thank Professor Yang for helping me figure out my true reason.

I would like to extend my sincere thanks to Professor Aylin Yener, Professor Nilanjan Ray Chaudhuri, and Professor Hui Yang for agreeing to serve on my dissertation committee and for their valuable comments and suggestions. Special thanks to Professor Aylin Yener for her support during the semester when I worked with her as her teaching assistant.

I also had the great pleasure of working with Professor Jingxian Wu, Xianwen Wu, and Zuoen Wang. I am grateful to all friends I shared an office with, including Chao Gan, Songtao Feng, Shiyang Leng, Omar Sleem, Malak Shah, Jiayu Mao and many others.

I am thankful to all of my friends at State College. I want to thank my roommates Lin Qiu and Yujia Wang. Thanks also to my friends Jia Zhu, Sheng Zou, Yi Zhang, Junyi Geng, Xuelu Li, Guoxiang Zhao and all my basketball teammates.

Last, but not least, I would like to thank my family. My parents have been forever supportive and encouraging in all of my pursuits.

Funding Acknowledgment and Disclaimer: This material is based upon work supported by the National Science Foundation (NSF) under Grants ECCS-1405403 and ECCS-1650299. Any opinions, findings, and conclusions or recommendations expressed in this dissertation are those of the author and do not necessarily reflect the views of the National Science Foundation.

Dedication

I dedicate this dissertation to

my family

for their constant support and unconditional love.

Chapter 1

Graph-based Multiple-Line Outage Identification

1.1 Introduction

Fast and accurate power line outage identification is critical for the prevention of cascading failures, real-time contingency analysis, and prompt and effective restoration of the power system. A plethora of power line outage identification methods have been proposed in the literature. In [4, 5], Tate and Overbye utilize the pre-outage network topology information and real-time phasor measurements to detect single and double line outages. In [6], Abdelaziz et al. introduce the support vector machine (SVM) technique from machine learning to detect single line outages. A mixed-integer programming approach is proposed by Emami and Abur in [7] to deal with single line outages. Essentially, these methods formulate line outage identification as a combinatorial problem, and solve it by exhaustively searching all

possible post-outage topologies. Due to the exponentially increasing complexity, such methods can only handle single or, at most, double line outages. The urge to cope with multiple line outages has motivated several existing works [8–11]. In [8, 9], the authors consider the calculation of line outage distribution factors in multiple-line outages case. In [11], He et al. adopts a Gaussian graphical model based approach to identify multiple-line outages at an affordable complexity, under the assumption that phasor angle measurements are conditionally independent with each other. In [10], Zhu and Giannakis formulate the power line outage identification problem as a sparse vector recovery problem, which is then solved efficiently by taking advantage of the recent progress in compressive sensing and variable selection. This approach has inspired a series of works, such as [12–14].

In this work, we study the multiple line outage identification problem by taking the cluster structural feature of multiple line outages into consideration. According to the outage records in [15], when multiple power line outages occur simultaneously, they tend to cluster in a local neighborhood. This can be intuitively explained in two ways. First, environment factors, such as lightning, wind, and snow, usually affect the transmission lines locally; Second, when outage occurs on certain transmission lines, the lines in conjunction with them usually have to bear more disturbance when power flow is automatically redistributed, thus are more susceptible to subsequent failures. Recent advances in sparse recovery indicate that exploiting structured properties in the sparse support of the underlying signal can significantly improve

the efficiency, accuracy, and computational complexity of baseline algorithms that do not take such structural features into consideration [16, 17]. This motivates us to develop new power line outage identification algorithms to exploit such cluster structure of outages to achieve efficient and accurate line outage identification.

We first formulate the problem as a sparse *binary*-valued vector estimation problem. Then, we propose three graph-based algorithms with reasonable computational complexity to solve it, namely, the graph-based Orthogonal Matching Pursuit (OMP) algorithm, the graph-based Compressive Sampling Matching Pursuit (CoSaMP) algorithm, and the graph-based Least Absolute Shrinkage and Selection Operator (LASSO) algorithm. All of the algorithms are analyzed and evaluated through simulated tests on IEEE-118 and 2383 bus systems. Simulation results confirm that the graph-based approaches are more efficient than baseline algorithms that do not exploit the cluster structure of multiple-line outages.

We adopt the following set of notations. We use bold-face lower-case, e.g., $\mathbf{x}$, and bold-face upper-case, e.g., $\mathbf{X}$, to denote vectors and matrices, respectively. $\|\mathbf{x}\|_p$ denotes the ℓ_p-norm of $\mathbf{x}$. Sets and events are denoted with calligraphic font (e.g., $\mathcal{T}$). The cardinality of a finite set $\mathcal{T}$ is denoted as $|\mathcal{T}|$.

1.2 System Model and Problem Formulation

We consider a power transmission network consisting of N buses and L lines. We use a graph $G = (\mathcal{N}, \mathcal{L})$ to represent it, where the node set $\mathcal{N} := \{1, ..., N\}$ denotes

the buses and the edge set $\mathcal{L} := \{(m,n)\} \subset \mathcal{N} \times \mathcal{N}$ represents the lines. The system monitors the voltage phasor angles and injected real power of all buses. Denote p_m as the real power injection on bus m, p_{mn} as the real power flow from bus m to bus n, θ_m as the voltage phasor angle of bus m, x_{mn} as the reactance of the line between bus m and n, which is equal to x_{nm}. Then under the DC power flow model, the instant power flow must satisfy [18]:

$$p_m = \sum_{n \in \mathcal{N}_m} p_{mn} = \sum_{n \in \mathcal{N}_m} \frac{\theta_m - \theta_n}{x_{mn}} \tag{1.1}$$

where $\mathcal{N}_m := \{m : (m,n) \in \mathcal{L}\}$, i.e., the set of buses connected with bus m in the graph G. Eqn. (1.1) can be equivalently expressed as

$$\mathbf{p} = \mathbf{B}\boldsymbol{\theta} \tag{1.2}$$

where $\boldsymbol{\theta} \triangleq [\theta_1, \theta_2 \ldots, \theta_N]^T \in \mathbb{R}^N$, $\mathbf{p} \triangleq [p_1, p_2 \ldots, p_N]^T \in \mathbb{R}^N$, $\mathbf{B}$ is a $N \times N$ matrix whose (m,n)-th entry is defined as

$$\mathbf{B}_{mn} = \begin{cases} -1/x_{mn} & \text{if } (m,n) \in \mathcal{L} \\ \Sigma_{v \in \mathcal{N}_m} 1/x_{mv} & \text{if } m = n \\ 0 & \text{otherwise} \end{cases} \tag{1.3}$$

We note that $\mathbf{B}$ is uniquely determined by the topology and line reactance parameters of the power network. Each transmission line $(m,n) \in \mathcal{L}$ is related to four

non-zero entries in $\mathbf{B}$, namely, $\mathbf{B}_{mn}$, $\mathbf{B}_{nm}$, $\mathbf{B}_{mm}$, and $\mathbf{B}_{nn}$.

When line outages happen in the system, the outaged lines are removed from the transmission network, resulting in a different matrix $\mathbf{B}'$. We assume that the power system works at a quasi-stable state during the events, i.e, the change of power injection $\mathbf{p}$ and load profile between the pre-event and post-event can be neglected under the timescale of outage events [4]. We also assume that the outage does not cause any islanding in the power grid. As a result of the line outage, voltage phasor angles will change automatically as the injected power is redistributed across the network to achieve another power flow balance. Denoting $\boldsymbol{\theta}' \triangleq [\theta'_1, \theta'_2 \ldots, \theta'_N]^T \in \mathbb{R}^N$ as the post-event phase angle measurements. Then, we have

$$\mathbf{B}\boldsymbol{\theta} = \mathbf{B}'\boldsymbol{\theta}' + \boldsymbol{\epsilon} \tag{1.4}$$

where, $\boldsymbol{\epsilon} \in \mathbb{R}^N$ consists of N i.i.d Gaussian random variables with zero mean and σ^2 variance. We use it to account for the small perturbations between the pre- and post-event injected powers. Our objective is to identify $\mathbf{B}'$ based on the knowledge of $\mathbf{B}$ and measurements $\boldsymbol{\theta}$ and $\boldsymbol{\theta}'$, and unveil the the outage locations based on the difference between $\mathbf{B}'$ and $\mathbf{B}$.

We follow the approach in [10] to reformulate (1.4) as a sparse vector recovery problem by representing matrix $\mathbf{B}$ in the form of a weighted Laplacian matrix. Denote l as the index of line (m, n), $\mathbf{m}_l \in \mathbb{R}^N$ as a column vector with all the entries

equal to 0 except the m-th and n-th elements, which are 1 and -1, respectively. Define $\mathbf{M} \triangleq [\mathbf{m}_1, \mathbf{m}_2, \ldots, \mathbf{m}_L] \in \mathbb{R}^{N\times L}$, $\mathbf{D} \triangleq \text{diag}\{\frac{1}{x_1}, \frac{1}{x_2}, \ldots, \frac{1}{x_L}\} \in \mathbb{R}^{L\times L}$. Then, we have

$$\mathbf{B} = \sum_{l=1}^{L} \frac{1}{x_l} \mathbf{m}_l \mathbf{m}_l^T = \mathbf{M}\mathbf{D}\mathbf{M}^T \tag{1.5}$$

Substituting (1.5) into (1.4), we have

$$\mathbf{B}\Delta\boldsymbol{\theta} = (\mathbf{B} - \mathbf{B}')\boldsymbol{\theta}' + \boldsymbol{\epsilon} \tag{1.6}$$

$$= \sum_{l\in\mathcal{L}'} \frac{1}{x_l} \mathbf{m}_l \mathbf{m}_l^T \boldsymbol{\theta}' + \boldsymbol{\epsilon} \tag{1.7}$$

$$= \sum_{l\in\mathcal{L}'} \frac{\mathbf{m}_l^T \boldsymbol{\theta}'}{x_l} \mathbf{m}_l + \boldsymbol{\epsilon} \tag{1.8}$$

where $\Delta\boldsymbol{\theta} = \boldsymbol{\theta}' - \boldsymbol{\theta}$, $\mathcal{L}'$ is the set of outaged lines. Eqn. (1.7) follows from the fact that, with the weighted Laplacian representation in (1.5), $\mathbf{B} - \mathbf{B}'$ can be expressed as the weighted summation of the matrices related to the outaged lines only.

Define $\mathbf{A} \triangleq [\frac{\mathbf{m}_1^T\boldsymbol{\theta}'}{x_1}\mathbf{m}_1, \frac{\mathbf{m}_2^T\boldsymbol{\theta}'}{x_2}\mathbf{m}_2, \ldots, \frac{\mathbf{m}_L^T\boldsymbol{\theta}'}{x_L}\mathbf{m}_L] \in \mathbb{R}^{N\times L}$, $\boldsymbol{\beta} \triangleq [\beta_1, \beta_2 \ldots, \beta_L]^T \in \mathbb{R}^L$, $\mathbf{y} \triangleq \mathbf{B}\Delta\boldsymbol{\theta}$, where

$$\beta_l = \begin{cases} 1 & \text{if } l \in \mathcal{L}' \\ 0 & \text{otherwise} \end{cases} \tag{1.9}$$

Then, (1.8) can be equivalently expressed as

$$\mathbf{y} = \mathbf{A}\boldsymbol{\beta} + \boldsymbol{\epsilon} \tag{1.10}$$

where $\mathbf{y}$ is the observation vector, which can be obtained based on $\mathbf{B}$ and the pre- and post-event phasor angle measurements, $\mathbf{A}$ is the measurement matrix, and $\boldsymbol{\beta}$ is a *binary*-valued vector to recover.

Assume that the number of the outaged lines is upper bounded by s, which is a small number compared with L. Then, the line outage identification problem can be formulated as a sparse recovery problem as follows:

$$\boldsymbol{\beta}^* = \arg \min_{\boldsymbol{\beta} \in \{0,1\}^L, \|\boldsymbol{\beta}\|_1 \leq s} \|\mathbf{y} - \mathbf{A}\boldsymbol{\beta}\|_2 \tag{1.11}$$

Essentially, (1.11) is an integer programming problem, and its complexity scales exponentially in s. To overcome the combinatorial complexity, in the following, we leverage the recent progress on sparse recovery and the power network topological information to develop two algorithms to solve (1.11) efficiently.

1.3 Graph-based Orthogonal Matching Pursuit

Orthogonal Matching Pursuit (OMP) is a widely adopted iterative greedy algorithm for sparse recovery. The algorithm works as follows: In each iteration, it greedily

selects the column which is most correlated with the current residual vector from the unselected columns of the measurement matrix. This column is then added into the set of selected columns. The algorithm then updates the residual vector by projecting the observation vector onto the linear subspace spanned by the selected columns. The algorithm iterates until the residual error is within a desired tolerance level. Compared with other alternative methods, a major advantage of the OMP is its simplicity and fast implementation.

In this section, we modify the OMP algorithm by incorporating the binary constraint on $\boldsymbol{\beta}$, and the topology information of the power system. We add graph G as an input to the algorithm. In each iteration, we keep track of two sets of lines, $\mathcal{S}^k$ and $\mathcal{L}^k$. $\mathcal{S}^k$ is used to track the set of candidate columns that would be considered in the step of greedy selection, with its initial value set to be the full set including all columns of $\mathbf{A}$. The estimate outage set $\mathcal{L}^k$ is used to track the set of identified outaged lines, with its initial value to be empty. The algorithm works as follows. At the beginning of the k-th iteration, we greedily select the column from $\mathcal{S}^{k-1}$ to minimize the ℓ_2 norm of the residual vector, and update the estimate outage set $\mathcal{L}^k$ by adding the new selected column index to $\mathcal{L}^{k-1}$. We point out that this index also correspond to a line in graph G. Next, we take the binary value constraint on $\boldsymbol{\beta}$ into consideration. We perform a Least Square Estimation (LSE) by restricting the values of $\beta_i, \forall i \in \mathcal{L}^k$ to be in $[0, 1]$ and letting the entries outside $\mathcal{L}^k$ to be zero. After obtaining $\boldsymbol{\beta}^k$, we force all of its entries to be binary through

thresholding. Next, we update the estimate outage set $\mathcal{L}^k$ again by including the support of the thresholded $\boldsymbol{\beta}^k$ only. The residual vector $\mathbf{r}^k$ is updated accordingly. If $\|\mathbf{r}^k\|_2$ is still above the tolerance level, the algorithm will take the power network topology into consideration. Since multiple line outages often happen in clusters, if $\mathcal{L}^k$ correctly identifies some of the outaged lines, then, intuitively, the rest of the outaged lines will be in close proximity to those included in $\mathcal{L}^k$. This motivates us to update $\mathcal{L}^k$ by including the lines that share buses with those included in $\mathcal{L}^k$ only. The algorithm then move to the next iteration until a desired tolerance level is achieved, or until it reaches the maximum number of iterations. We summarize the algorithm in Algorithm 1.

Algorithm 1 (Graph OMP) Input: $\mathbf{y}$, $\mathbf{A}$, graph G, threshold τ. Output $\boldsymbol{\beta}$.

Initialize: $\mathbf{r}^0 = \mathbf{y}, \boldsymbol{\beta}^0 = \mathbf{o}, \mathcal{S}^0 = \mathcal{L}$, $\mathcal{L}^0 = \emptyset$, $k = 1$.
while $k \leq k_{max}$ or $\|\mathbf{r}^k\|_2 \leq \delta$ **do**
    $l^k := \arg\min_{l \in \mathcal{S}^{k-1}} \|\mathbf{r}^{k-1} - \mathbf{A}(:, j)\|_2^2$
    $\mathcal{L}^k := \mathcal{L}^{k-1} \cup \{l^k\}$
    Estimate $\boldsymbol{\beta}^k := \arg\min \|\mathbf{r}^{k-1} - \mathbf{A}\boldsymbol{\beta}\|_2^2$ s.t. $\beta_i \in [0, 1]$ for $i \in \mathcal{L}^k$, and $\beta_i = 0$, $\forall i \notin \mathcal{L}^k$
    Update $\mathcal{L}^k := \{l : \beta_l^k \geq \tau\}$
    Update residual vector $\mathbf{r}^k := \mathbf{r}^{k-1} - \sum_{j \in \mathcal{L}^k} \mathbf{A}(:, j)$
    if $\|\mathbf{r}^k\|_2 \leq \delta$ **then return**
    else
        Update $\mathcal{S}^k$ by adding all lines sharing buses with the lines included in $\mathcal{L}^k$;
        $k = k + 1$;
    end if
end while

1.4 Graph-based CoSaMP

CoSaMP is an algorithm based on OMP. It uses an approach inspired by the restricted isometry property (RIP) [19] to accelerate the algorithm and to provide strong guarantees that OMP cannot. Specifically, if the measurement matrix $\mathbf{A}$ satisfies the RIP, the vector $\mathbf{A}^T\mathbf{A}\boldsymbol{\beta}$ can serve as a proxy for the sparse signal $\boldsymbol{\beta}$, since the largest components of the proxy point to the largest components of the original signal $\boldsymbol{\beta}$ with high probability. In each iteration of CoSaMP, it first identifies the largest components from the proxy of the residual, merges them into the set of selected component, and then solves a least-squares problem to approximate the target signal on the merged set of components. The algorithm then performs a pruning step to produce a new approximation by retaining only the largest entries in the least-squares approximation. The residual is then updated and fed to the next iteration until a stopping condition is satisfied.

In [20], Hegde et. al propose a Graph-based CoSaMP algorithm to account for sparsity structures defined via graphs. They introduce a novel general sparsity model called weighted graph model (WGM). Specifically, they define $G = (\mathcal{V}, \mathcal{E})$ as an undirected, weighted graph where the node set $\mathcal{V}$ corresponds to the unknown sparse vector $\boldsymbol{\beta}$, and each edge has a real-valued weight. In order to control the sparsity patterns, the WGM offers three parameters on the desired supports $\mathcal{S} \subseteq \mathcal{V}$, i.e., the sparsity level s; the maximum number of connected components (trees)

corresponding to $\mathcal{S}$, denoted as g; and the bound on the total weight of edges in the forest corresponding to $\mathcal{S}$, denoted as B. By choosing different edge weights and those three parameters, different sparsity structures can be encoded with the WGM. Essentially, the WGM captures sparsity structures with a small number of trees in G.

The algorithmic core of the graph-CoSaMP algorithm is a computationally efficient procedure to project an arbitrary vector into the WGM. Mathematically, for any given vector $\boldsymbol{\beta}$, the algorithm looks for the best vector $\boldsymbol{\beta}'$ whose support is compliant with the sparsity structure encoded by the WGM to minimize $\|\boldsymbol{\beta} - \boldsymbol{\beta}'\|^2$. However, searching for the best $\boldsymbol{\beta}'$ is a NP-hard combinatorial optimization problem [21]. In order to reduce the computational complexity, the framework propose two approximate model-projection algorithms, namely tail approximation and head approximation, to obtain a sub-optimal $\boldsymbol{\beta}'$ with complementary approximation guarantees. The approximation algorithms are based on a connection to the prize-collecting Steiner Forest problem (PCSF).

For the completeness of this work, in the following, we will first introduce the PCSF problem, and then discuss how our problem can be related to PCSF.

1.4.1 The Prize Collecting Steiner Forest (PCSF) Problem

PCSF is a variant of the Prize Collecting Steiner Tree (PCST) problem. Let $G = (\mathcal{V}, \mathcal{E})$ be an undirected, weighted graph with edge costs $c(e) \in \mathbb{R}, \forall e \in \mathcal{E}$ and node

prizes $\pi(v) \in \mathbb{R}, \forall v \in \mathcal{V}$. For a subset of edges $\mathcal{E}' \subset \mathcal{E}$, its cost $c(\mathcal{E}') = \sum_{e \in \mathcal{E}'} c(e)$. Similarly, for a subset of nodes $\mathcal{V}' \subset \mathcal{V}$, its prize $\pi(\mathcal{V}') = \sum_{v \in \mathcal{V}'} \pi(v)$. We also denote $\bar{\mathcal{V}}'$ as the complement of $\mathcal{V}'$. Then the goal of the PCST problem is to find a subtree $\mathcal{T} = (\mathcal{V}', \mathcal{E}')$ to minimize $c(\mathcal{T}) + \pi(\bar{\mathcal{T}})$.

This problem is a generalization of the classical Steiner tree problem [22], and it is also NP-hard. In the seminal work of Goemans and Williamson (GW) [23], a primal-dual algorithm is developed with certain approximation guarantee.

The objective of PCSF is to find a subgraph $\mathcal{F} = (\mathcal{V}', \mathcal{E}')$ that minimizes $c(\mathcal{F}) + \pi(\bar{\mathcal{F}})$ with at most g connected components (trees) in $\mathcal{F}$. In [20], a fast GW algorithm is proposed, which returns a forest with the following guarantee:

$$c(\mathcal{F}) + 2\pi(\bar{\mathcal{F}}) \leq 2c(\mathcal{F}_{OPT}) + 2\pi(\bar{\mathcal{F}}_{OPT}) \tag{1.12}$$

where $\mathcal{F}_{OPT}$ is the optimal solution of the PCSF problem.

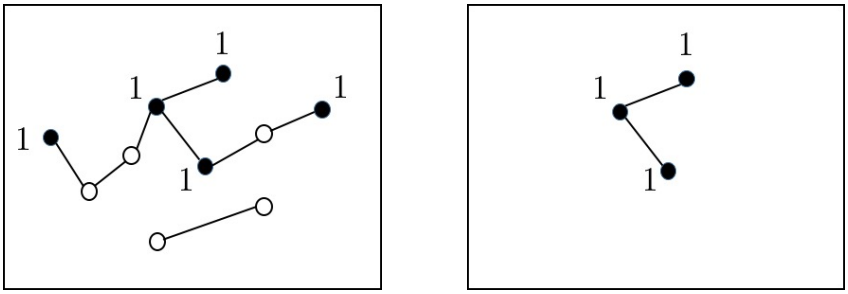
By setting $\pi(i) = \beta_i^2$ and $c(e) = w(e) + 1$, we have $c(\mathcal{F}) = w(\mathcal{F}) + (|\mathcal{F}| - g)$, $\pi(\bar{\mathcal{F}}) = \|\boldsymbol{\beta} - \boldsymbol{\beta}_{\mathcal{F}}\|^2$. After multiplying the edge costs with a parameter λ, the PCSF objective function essentially becomes a Lagrangian relaxation of the WGM-constrained optimal projection problem. The tail and head approximation algorithms in [20] adaptively search for the optimal λ to ensure that the PCSF-GW algorithm returns a good approximation of $\boldsymbol{\beta}$ with theoretical guarantees.

1.4.2 Graph CoSaMP Algorithm

In this subsection, we adapt the Graph CoSaMP proposed in [20] for the multiple-line outage identification problem. For the sparse recovery problem in (1.11), the unknown vector $\boldsymbol{\beta}$ represents the statuses of power lines. The corresponding WGM graph $M = (\mathcal{V}, \mathcal{E})$ is defined as follows. First, we define the node set $\mathcal{V}$ as the set of the power lines. If two power lines are connected through a bus in the power grid, there is an edge connecting the corresponding nodes in M. We let the node prize $\pi(i) = \beta_i^2$, and the edge cost $c(e) = c$. As we will explain later, we fix c be to a constant in $(0.5, 1)$.

The adapted Graph CoSaMP is summarized in Algorithm 2. Specifically, after forming a proxy of the residue vector, we adopt the tail approximation algorithm in [20] to find an approximation of the proxy with the desired sparse forest structure. After we merge the support of the returned forest with the support from previous step, we update the estimation of the signal vector by performing LSE over the support under the constraint that $\beta_i \in [0, 1]$, and then threshold it to make it binary. In the pruning step, we use PCSF-GW algorithm to get an approximated signal vector whose support has the desired forest structure.

The input of PCSF-GW comes from the thresholded LSE result over the support $\mathcal{Z}^k$. Therefore, the node prizes can only be zero or one, as shown in Fig. 1.1(a). We use black dots to represent the nodes with prize one. Consider the case where g is set to be one. If the edge cost is greater than one, PCSF-GW will select only one



(a) Input of PCSF. (b) Output of PCSF.

Figure 1.1: A pruning example of PCSF with $g = 1$.

node with prize one to optimize the objective function. On the other hand, if the edge cost is very small, all nodes with prize one will be connected by edges to form one cluster. In order to obtain the desired cluster structure shown in Fig. 1.1(b), we choose a constant edge cost $c \in (0.5, 1)$.

Algorithm 2 (Graph CoSaMP) Input: $\mathbf{y}$, $\mathbf{A}$, s, g, M, τ. Output: $\boldsymbol{\beta}$.

Initialize: $\mathbf{r}^0 = \mathbf{y}, \boldsymbol{\beta}^0 = \mathbf{o}, \mathcal{S}^0 = \mathcal{L}$, $\mathcal{L}^0 = \emptyset$, $k = 1$.
while Stopping criterion is not met **do**
    Form a proxy $\mathbf{t} = \mathbf{A}^\intercal \times \mathbf{r}^{k-1}$
    Identify support $\mathcal{T} = \text{TailApprox}(M, \mathbf{t}, s, g)$
    Merge $\mathcal{Z}^k = \mathcal{T} \cup \text{supp}(\boldsymbol{\beta}^{k-1})$
    Estimate $\boldsymbol{\beta}^k = \arg\min \| \mathbf{b} - \mathbf{A}\boldsymbol{\beta} \|_2$ s.t. $\text{supp}(\boldsymbol{\beta}) \in \mathcal{Z}^k$, $\beta_i \in [0, 1]$
    if $\beta_i^k \geq \tau$ **then** $\beta_i^k = 1$
    else $\beta_i^k = 0$
    end if
    Prune $\boldsymbol{\beta}^k = \text{PCSF-GW}(M, \boldsymbol{\beta}^n, g)$
    $\mathbf{r}^{k+1} = \mathbf{y} - \mathbf{A}\boldsymbol{\beta}^k$
    $k = k + 1$
end while
return $\boldsymbol{\beta}$

1.5 LASSO Based Algorithms

Least Absolute Shrinkage and Selection Operator (LASSO) is another commonly used method in sparse approximation problem besides OMP. It was first formulated by Tibshirani [24] for least squares models. LASSO promotes the sparsity of the solution by regularizing the least square error with a constraint on the ℓ_1 norm of the solution. In stead of projecting the residual on the subspace spanned by some columns of the measurement matrix $\mathbf{A}$ in OMP, LASSO relaxes the sparse recovery problem to a convex optimization problem which can be effectively solved by existing optimizations tools [25]. In [10], LASSO was adopted to solve the line outage identification problem. In this section, we adapt traditional LASSO by imposing the binary constraint on $\boldsymbol{\beta}$ and incorporating the cluster feature of the outaged lines. We introduce two approaches, namely, Binary LASSO and Graph LASSO in the following.

1.5.1 Binary LASSO

With the binary constraint on $\boldsymbol{\beta}$, the aforementioned equivalent representation (1.11) can be formulated as a LASSO problem with a binary constraint on $\boldsymbol{\beta}$:

$$\min_{\boldsymbol{\beta}\in\{0,1\}^L} \|\mathbf{y} - \mathbf{A}\boldsymbol{\beta}\|_2^2 + \lambda\|\boldsymbol{\beta}\|_1. \tag{1.13}$$

However, (1.13) is known to be NP-hard in general. Fortunately, there exist

many approximation methods to solve the binary LASSO problem [26]. One popular approach is through semidefinite relaxation (SDR) [26].

Let $\boldsymbol{\beta} \triangleq \frac{\boldsymbol{\theta}+\mathbf{e}}{2}$, where $\mathbf{e}$ is a column vector of all ones. Then, (1.13) can be equivalently expressed as

$$\min_{\boldsymbol{\theta}\in\{-1,1\}^L} \|\mathbf{y}_0 - \mathbf{A}\boldsymbol{\theta}\|_2^2 + \lambda\|\boldsymbol{\theta} + \mathbf{e}\|_1, \tag{1.14}$$

where $\mathbf{y}_0 = 2\mathbf{y} - \mathbf{A}\mathbf{e}$.

Then, (1.14) is equivalent to

$$\begin{aligned} \min_{\mathbf{X},\boldsymbol{\theta}} \quad & \mathrm{Tr}(\mathbf{A}^\intercal\mathbf{A}\mathbf{X}) - 2\mathbf{y}_0^\intercal\mathbf{A}\boldsymbol{\theta} + \mathbf{y}_0^\intercal\mathbf{y}_0 + \lambda\|\boldsymbol{\theta} + \mathbf{e}\|_1 \\ \text{s.t.} \quad & \mathbf{X} = \boldsymbol{\theta}\boldsymbol{\theta}^\intercal, \mathbf{X}_{ii} = 1, i = 1, \ldots, n. \end{aligned} \tag{1.15}$$

Since the constraint $\mathbf{X} = \boldsymbol{\theta}\boldsymbol{\theta}^\intercal$ is non-convex, we use SDR to relax it to $\mathbf{X} \succeq \boldsymbol{\theta}\boldsymbol{\theta}^\intercal$, which is convex and can be solved as an semidefinite program problem:

$$\begin{aligned} \min_{\mathbf{X},\boldsymbol{\theta}} \quad & \mathrm{Tr}(\mathbf{A}^\intercal\mathbf{A}\mathbf{X}) - 2\mathbf{y}_0^\intercal\mathbf{A}\boldsymbol{\theta} + \mathbf{y}_0^\intercal\mathbf{y}_0 + \lambda\|\boldsymbol{\theta} + \mathbf{e}\|_1 \\ \text{s.t.} \quad & \begin{bmatrix} \mathbf{X} & \boldsymbol{\theta} \\ \boldsymbol{\theta}^\intercal & 1 \end{bmatrix} \succeq 0, \mathbf{X}_{ii} = 1, i = 1, \ldots, n \end{aligned} \tag{1.16}$$

It is shown that the SDR is tight and has good approximate solutions under

certain assumptions [26]. We note that $\theta \in \{-1, 1\}$ in (1.15) due to the equality constraint, but is relaxed into $[-1, 1]$ in (1.16). Thus, we round the output to recover to the original solution.

1.5.2 Graph LASSO

In this subsection, with the same motivation as Graph OMP, we modify the binary LASSO by incorporating the structured properties of the outage pattern into the formulation. To take advantage of the structured information, we utilize the matrix $\mathbf{M}$ defined in Section 1.2 to encode the topological information of the power system. Specifically, we take the absolute value of the elements in $\mathbf{M}$ and obtain another matrix $\mathbf{M}'$. Then, Graph LASSO imposes a penalty term that encourages cluster features in the solution, as follows:

$$\min_{\beta \in \{0,1\}^L} \|\mathbf{y} - \mathbf{A}\boldsymbol{\beta}\|_2^2 + \lambda \sum_{i=0}^{N} \|\mathbf{M}'(i,:) \odot \boldsymbol{\beta}\|_2, \tag{1.17}$$

where $\odot$ denotes element-wise multiplication, and $\mathbf{M}'(i,:)$ denotes the i^{th} row of matrix $\mathbf{M}'$. Then, with the similar idea in Binary LASSO, we use SDR and have the following approximation:

$$\begin{aligned} &\min_{\mathbf{X},\boldsymbol{\theta}} \mathrm{Tr}(\mathbf{A}^\intercal \mathbf{A}\mathbf{X}) - 2\mathbf{y}_0^\intercal \mathbf{A}\boldsymbol{\theta} + \mathbf{y}_0^\intercal \mathbf{y}_0 + \lambda \sum_{i=0}^{N} \|\mathbf{M}'(i,:) \odot (\boldsymbol{\theta} + \mathbf{e})\|_2. \\ &\text{s.t.} \quad \begin{bmatrix} \mathbf{X} & \boldsymbol{\theta} \\ \boldsymbol{\theta}^\intercal & 1 \end{bmatrix} \succeq 0, \mathbf{X}_{ii} = 1, i = 1, \ldots, n \end{aligned} \tag{1.18}$$

We note that $\lambda \sum_{i=0}^{N} \|\mathbf{M}'(i,:) \odot (\boldsymbol{\theta} + \mathbf{e})\|_2$ is convex in $\boldsymbol{\theta}$ as $\boldsymbol{\theta}$ is relaxed into $[-1, 1]$ after SRR. Thus (1.18) can be solved by many exist convex solvers.

To see why adding the penalty term $\sum_{i=0}^{N} \|\mathbf{M}'(i,:) \odot \boldsymbol{\beta}\|_2$ could encourage the desired cluster structure in the optimal solution, we use the 6-bus system in Fig. 1.2 as an example. The 6-Bus system consists of 6 buses and 7 lines. The topology of the system can be represented by matrix $\mathbf{M}'$ in Fig. 1.3, where, each column and each row represent a transmission line and a bus, respectively. The non-zero entries in a column correspond to the terminal buses of that transmission line. Thus, for all lines that are connected through a shared bus i, the i-th entry of the corresponding columns are equal to one. Then, for all $\boldsymbol{\beta}_s$ that have the same sparsity, the outage patterns with cluster structure tend to incur a lower penalty. For example, if $s = 2$, $\boldsymbol{\beta}$ has two non-entries which represents two outaged lines. If these two outaged lines are not clustered, e.g., line 1 and line 5, then there are four non-zero vectors after the element-wise multiplication which are $\mathbf{M}'(1,:) \odot \boldsymbol{\beta} = [1,0,0,0,0,0,0]^\intercal$, $\mathbf{M}'(3,:) \odot \boldsymbol{\beta} = [0,0,0,0,1,0,0]^\intercal$, $\mathbf{M}'(4,:) \odot \boldsymbol{\beta} = [1,0,0,0,0,0,0]^\intercal$, and $\mathbf{M}'(5,:) \odot \boldsymbol{\beta} = [0,0,0,0,1,0,0]^\intercal$. If the two outaged lines are clustered, i.e., line 1 and line 2, then the non-zero groups are $\mathbf{M}'(1,:) \odot \boldsymbol{\beta} = [1,1,0,0,0,0,0]^\intercal$, $\mathbf{M}'(4,:) \odot \boldsymbol{\beta} = [1,0,0,0,0,0,0]^\intercal$, and $\mathbf{M}'(5,:) \odot \boldsymbol{\beta} = [0,1,0,0,0,0,0]^\intercal$. Thus, the penalty term $\sum_{i=0}^{N} \|\mathbf{M}'(i,:) \odot \boldsymbol{\beta}\|_2$ is lower for the second outage pattern. Essentially, for a given outage pattern $\boldsymbol{\beta}$, the outaged lines are grouped into clusters based on whether they share a bus or not, each term $\mathbf{M}'(i,:) \odot \boldsymbol{\beta}$ corresponds to a cluster

sharing bus i. Due to the convexity of the penalty function, the Graph LASSO tends to select outage patterns with more and larger clusters.

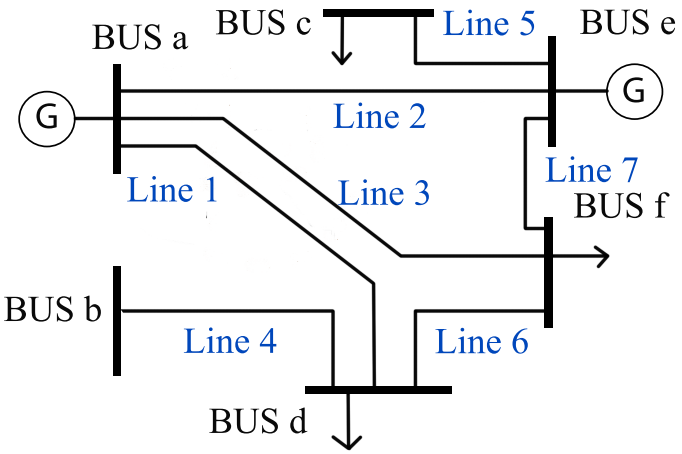


Figure 1.2: 6 buses power system.

	Line 1	Line 2	Line 3	Line 4	Line 5	Line 6	Line 7
Bus a	1	1	1	0	0	0	0
Bus b	0	0	0	1	0	0	0
Bus c	0	0	0	0	1	0	0
Bus d	1	0	1	1	0	1	0
Bus e	0	1	0	0	1	0	1
Bus f	0	0	0	0	0	1	1

Figure 1.3: M' of the 6 buses power system.

where, each column and each row represent a transmission line and a bus, respectively. The non-zero entries in a column correspond to the terminal buses of that transmission line. Thus, for all lines that are connected through a shared bus i, the i-th entry in the corresponding columns are equal to one. With a fixed sparsity level s, penalty term $\|\sqrt{\mathbf{M}'\boldsymbol{\beta}}\|_1$ encourages cluster structure in the solution.

Since $\mathbf{M}'\boldsymbol{\beta}$ calculates the addition of the column vectors of $\mathbf{M}'$ that represents the outaged lines, if more outaged lines share a bus, the correspond $\|\sqrt{\mathbf{M}'\boldsymbol{\beta}}\|_1$ will be smaller. For example, if $s = 2$, $\boldsymbol{\beta}$ has two non-entries which represents two outaged lines. If these two outaged lines are not clustered, e.g., line 1 and line 5, then $\mathbf{M}'\boldsymbol{\beta} = [1, 0, 1, 1, 1, 0]^\intercal$. If the two outaged lines are clustered, i.e., line 1 and line 2, then $\mathbf{M}'\boldsymbol{\beta} = [2, 0, 0, 1, 1, 0]^\intercal$. Thus, the penalty term $\|\sqrt{\mathbf{M}'\boldsymbol{\beta}}\|_1$ will encourage the optimizer to choose the second solution.

1.6 Simulation Results

In this section, we compare our proposed algorithms with baseline algorithms OMP, CoSaMP and LASSO on an IEEE 118-Bus system and an IEEE 2383-Bus system, respectively. The parameters of the systems are fully described in the toolbox of MATPOWER [27]. We simulate the line outages by eliminating some transmission lines in the power network. We collect pre- and post-event measurements of the voltage phasor angle of every bus after running the power flow function.

We consider the result returned by an algorithm a *success* if it matches with the set of lines removed from the system exactly. We randomly generate the outage patterns, and evaluate the performances of the algorithms by counting the ratio of success.

Test Case 1: First, we study the performance of the proposed algorithms on identifying a single cluster of line outages with complete measurements from all

the buses. The size of the cluster ranges from one to three lines. For the single line case, all of the 179 lines in the 118-bus system are tested and repeated for 10 times, totally 1790 cases. For the double and triple line clustered outage, we randomly choose 1500 cases. We set the standard deviation of the noise to be 0, 1%, 3%, and 5% of the injected power, and evaluate the performances of the algorithms at those noise levels. The percentage of successful identification is recorded and compared in Fig. 1.4. As we observe, Graph OMP has significant improvement over OMP for all three scenarios. Graph CoSaMP has better performance in multiple-line outage cases over CoSaMP, especially in the three-line outage case. Binary LASSO and Graph LASSO has better performance than conventional LASSO, especially in the low noise regime. Graph LASSO has similar performance as Binary LASSO in the high noise regime, but performs better in the low noise regime for the multiple-line outage cases. Overall, Graph CoSaMP and Graph LASSO achieve the best performances for the multiple-line outage cases, which indicates the effectiveness of utilizing structured sparsity for the outage identification problem.

Test Case 2: Then, we study the performances of the proposed algorithms with two clusters of line outages. We consider the case where both clusters consist of two outaged lines, and three outaged lines, respectively. The identification performances are shown in Fig. 1.5. Similar to Test Case 1, we note that Graph CoSaMP and Graph LASSO outperform the rest algorithms for the 3+3 line outage case, where all algorithms except OMP perform similarly for the 2+2 line outage case. The

results indicate that leveraging the structral properties is more advantages when the structure pattern is more prominent.

Test Case 3: To further illustrate the performances of the proposed algorithms, we use partial measurements to test the 3+3 clusters case with a fixed noise level of 1%. Phase angle measurements are collected from a randomly selected subset of the buses, where size of the the subset is set to be 100%, 90%, 80%, 70%, and 60% of the total number of buses, respectively. Fig. 1.6 shows the identification performances of the algorithms. We note that the proposed algorithms outperforms their the conventional counterparts, among which Graph CoSaMP and Graph LASSO still achieve the best performances.

Test Case 4: Next, we test the algorithms on the 2383-bus system, which is a large power system model coming from the Polish power system. The identification performance of the algorithms for 3+3 line outage case is presented in Fig. 1.7. Similar to the results for the 128-bus system, we note that Graph CoSaMP and Graph LASSO outperform the other algorithms, which indicates the efficiency of our proposed graph based algorithms on large-scale power systems.

Test Case 5: Finally, we compare the running times of the algorithms. All algorithms are run in MATLAB R2016a, on a MacBook Pro with a 2.2GHz CPU and a 16GB RAM. Table 1.1 shows the average running times of the algorithms. We note that Graph OMP has slight improvement over OMP on the average running time, while Graph CoSaMP takes much more time than other algorithms

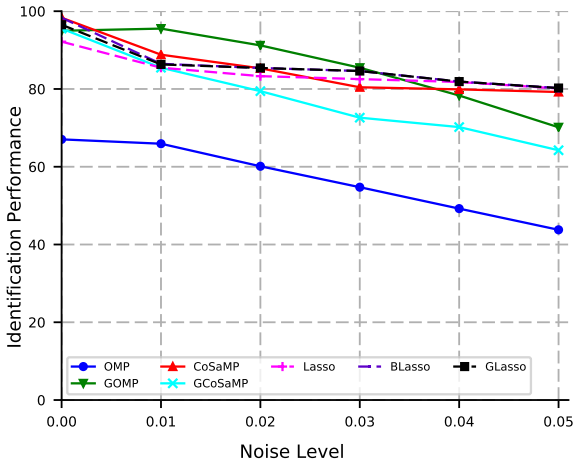
due to its increased complexity in the graph model-based projections in the head approximation and pruning steps.

Table 1.1: Average running times in seconds.

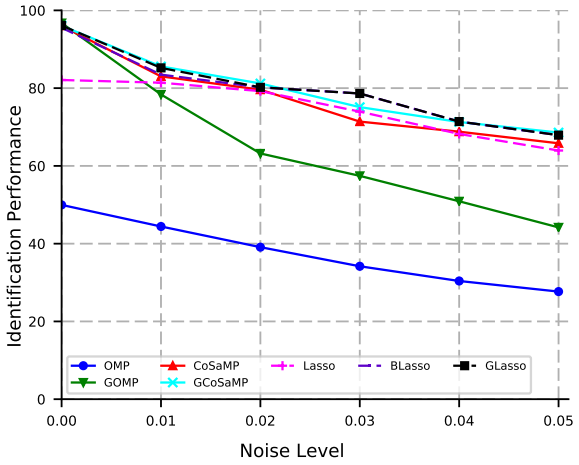
Power system	Outage type	OMP	Graph OMP	CoSaMP	Graph CoSaMP	Lasso	GLasso
128-bus	1 line	2e-3	1.8e-3	8e-4	1.60	1.66	1.81
	2 lines	2.4e-3	2.3e-3	6e-4	1.60	1.66	1.8
	3 lines	4.1e-3	3.9e-3	7e-4	1.82	1.72	1.85
	2+2 lines	3.7e-3	3.9e-3	7e-4	1.57	1.55	1.86
	3+3 lines	6.4e-3	6.5e-3	9e-4	1.75	1.65	1.88
2383-bus	2+2 lines in 2383-Bus	0.16	0.16	0.02	2.00	3.01	3.66
	3+3 lines in 2383-Bus	0.26	0.29	0.03	2.30	3.34	3.71

1.7 Conclusions

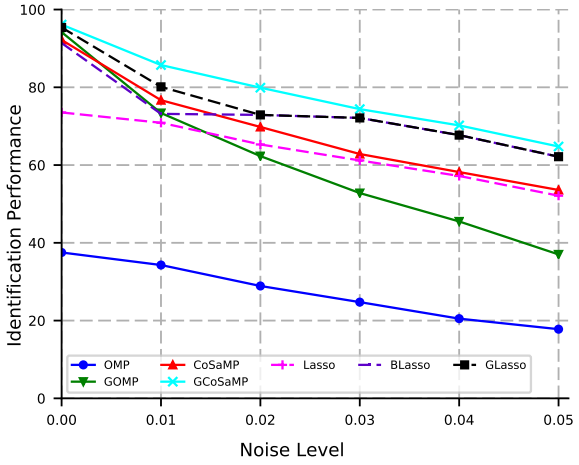
In this paper, we proposed several approaches to leverage the cluster feature of power line outages in transmission systems for multiple-line outage identification. Modeling the power network as an undirected graph, we proposed graph based algorithms, including Graph OMP, Graph CoSaMP and Graph LASSO, to utilize the topological information of the power system and obtain solutions with the desired cluster structure. Numerical tests demonstrated that the proposed algorithms outperform their conventional counterparts that do not exploit the structural properties of the line outage patterns.



(a) Single-line outage.

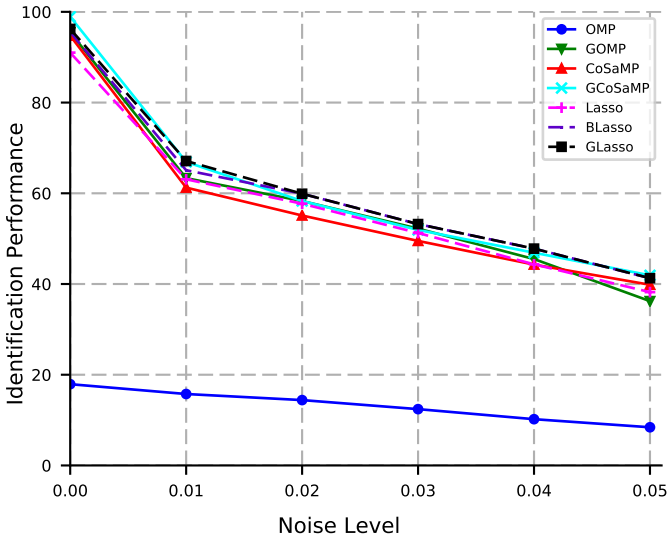


(b) Double-line outage.

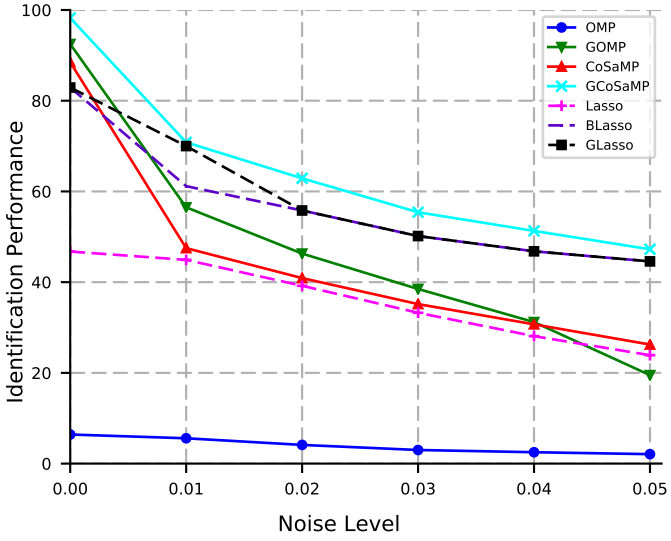


(c) Three-line outage.

Figure 1.4: Outage identification for a single cluster.



(a) 2+2 lines outage.



(b) 3+3 lines outage.

Figure 1.5: Outage identification for two clusters.

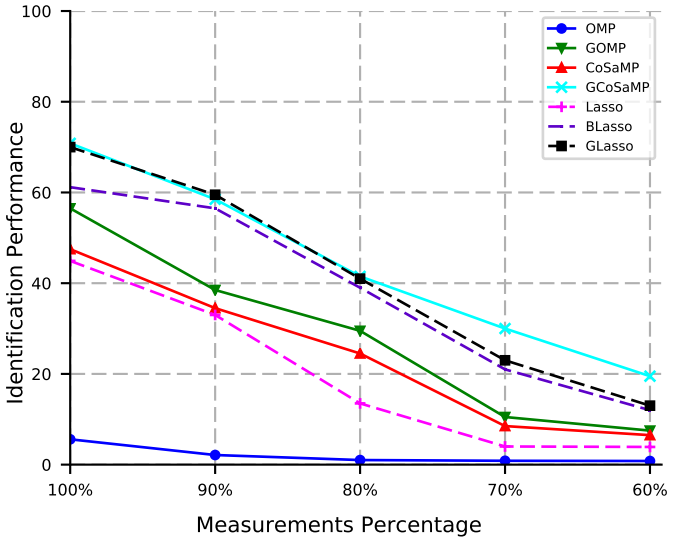


Figure 1.6: Outage identification with partial measurements.

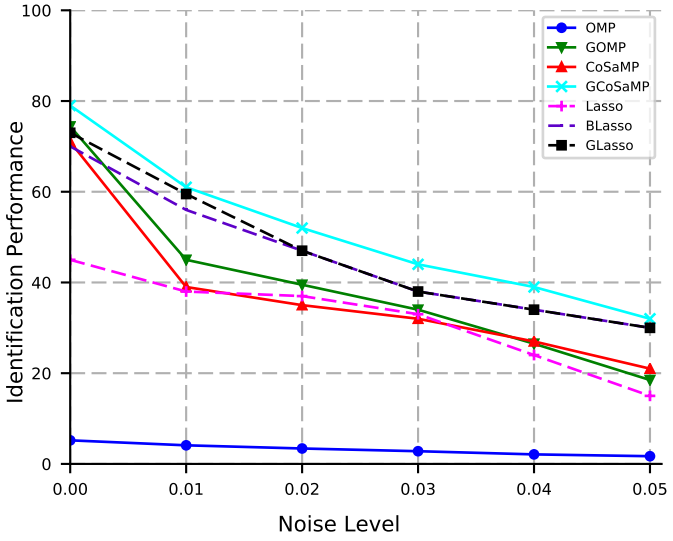


Figure 1.7: Outage identification for 3+3 clusters in 2383-Bus system.

Chapter 2

Time Series Prediction and Quick Anomaly Detection in Power Systems

In this chapter, we consider data-driven approach to detect anomaly events in power systems. Different from model-based approaches in Chapter 1, this data-driven approach does not require prior knowledge of the power system model parameters. Instead, it utilizes a Long Short Term Memory (LSTM) model to capture the state evolution of the power system. Due to the expressiveness of the LSTM model, it is able to track the system states with small prediction error when it operates under normal conditions. However, when the system is perturbed by certain events that cannot be predicted by the model, the prediction error will increase dramatically. Thus, by tracking the prediction error of the trained LSTM model, the data-driven online approach is able to detect events in a timely fashion. The event detection problem is then cast into the quick change detection framework, where a Cumulative Sum (CUSUM) based approach is proposed. To overcome the

difficulty that the statistics of the prediction error when events happen is generally unknown beforehand, a generalized likelihood ratio test (GLRT) is incorporated into the CUSUM procedure. A Rao-test is then adopted to reduce the computationally complexity of GLRT. Finally, the LSTM based event detection approach is validated with real-world PMU measurements.

2.1 Introduction

The ubiquitous uncertainties existing in power systems, such as line outage, short-circuit fault, load shedding, generation change, make the stable and reliable operation very challenging. Efficiently detecting those events is crucial for a power system to maintain its stable operation and prevent it from large scale blackouts or cascading failures [28].

On the other hand, the rapid deployment of advanced sensing devices in power systems, such as phasor measurements units (PMUs), makes real-time grid monitoring and control possible. Various online event detection methods have been proposed in recent years [29–34]. In order to reduce the detection delay, the problem is often cast into the quickest change detection (QCD) framework [35–38]. In [35], QCD is used to detect and identify transmission line outage in near real-time. References [36, 37] adapt the QCD framework to detect false data injection attacks in power grids. In [38], an S-transform based cumulative sum detector is proposed to detect unintentional islanding. In [39], QCD is adopted to monitor the power

quality in smart grids. These methods highly rely on either the accurate distributional information of the measurements, such as phase angle change in [35], or knowledge about the power system, such as the grid topology in [36, 37]. However, in practice, time series collected from power grids usually contain complex patterns that cannot be accurately modeled by independent and identically distributed (i.i.d.) random variables. Furthermore, prior knowledge of power system usually is not given. Thus, applying QCD methods directly on raw measurements may lead to degraded detection performance.

Recently, researchers have proposed to resolve this issue by combining time series prediction and event detection [33, 40, 41]. In [33], a state space model is first applied to predict future frequency measurements, and the probabilistic prediction errors are summarized to quantify the uncertainty. Specifically, a probability threshold and a duration threshold are combined to detect the events using the prediction error. In [40], a long short term memory (LSTM) network is used to predict the time sequence and the prediction error is modeled as a multivariate Gaussian random vector, which is then used to evaluate the likelihood of events. In [41], the normal behavior is first learned by a Kalman filter model, and then the difference between the observation and the predicted value from the model is used to detect events. While such prediction methods are able to capture the temporal dynamics of the power system, the corresponding event detection methods are usually heuristic and may not detect events timely.

In this chapter, we aim to incorporate time series prediction to the QCD framework. We first adopt an LSTM model to capture the temporal evolution of the measurements when the system operates under a normal condition, which can be trained using historical measurements. We then utilize the trained model to predict upcoming measurements and track the prediction error in real time. We assume that the prediction error before the change point follows a normal distribution, whose parameters can be learned based on training data. When an event happens and the system state is perturbed, the trained LSTM model cannot track the state change closely. Thus, the corresponding distribution of the prediction error will change dramatically. Such distribution shift enables us to cast the event detection problem into the QCD framework, which is then solved efficiently through a CUSUM based method.

There are three main advantages of the proposed LSTM based event detection approach:

First, it is purely data-driven and does not require any prior knowledge of the power system. Thus, it is a universal approach and can be deployed in various systems with complicated dynamics.

Second, the event detection method relies on classical approaches in QCD, thus is theoretically grounded and can achieve (near-)optimal tradeoff between the probability of false alarm and detection delay.

Third, the effectiveness of the proposed approach is validated on a real-world

dataset. It outperforms two baseline algorithms, yielding significantly shorter detection delay when the probability of false alarm is reasonably low.

2.2 Problem Formulation

In this chapter, we focus on data-driven online event detection in power system. We consider a discrete-time model, where measurements are collected by certain sensing devices at discrete time points $n = 1, 2, \ldots$. Let $\mathbf{x}_n \in \mathbb{R}^m$ be the m-dimensional measurement vector collected at time n. For example, $\mathbf{x}_n$ could include the voltage, current, etc., measured by a PMU. Due to the inherent temporal correlation existing in the power system, $\mathbf{x}_n$s are not i.i.d. random variables in general. Without loss of generality, we assume that the measurements collected at time n (i.e., $\mathbf{x}_n$) depends on the previous p measurements $\mathbf{x}_{n-p}, \cdots, \mathbf{x}_{n-1}$ through the following equation:

$$\mathbf{x}_n = g(\mathbf{x}_{n-p}, \mathbf{x}_{n-p+1}, \cdots, \mathbf{x}_{n-1}) + \boldsymbol{\epsilon}_n, \tag{2.1}$$

where $g(\cdot)$ is a function that captures the temporal dependency of the measurements, and $\boldsymbol{\epsilon}_n$ is a random vector that captures other randomness existing in the system. For systems with simple topologies and control space, $g(\cdot)$ may have an explicit form determined by the state transition of the system. However, in general, when the system becomes complicated, $g(\cdot)$ may not admit a clean closed form expression.

We assume $\boldsymbol{\epsilon}_n$ is independent with all previous measurements $\mathbf{x}_1, \ldots, \mathbf{x}_{n-1}$.

Besides, depending on whether there exists an event at time n, $\boldsymbol{\epsilon}_n$ may follow different distributions. Specifically, we assume that when the system operates under a normal condition, $\boldsymbol{\epsilon}_n$ is an i.i.d. Gaussian random vector with mean vector to be zero and covariance matrix Σ, i.e., $\boldsymbol{\epsilon}_n \sim \mathcal{N}(\mathbf{0}, \Sigma)$. However, if an event happens at time n, and the system encounters a large disturbance, the distribution of $\boldsymbol{\epsilon}_n$ will be shifted by an *unknown* constant $\boldsymbol{\mu}$, i.e., $\boldsymbol{\epsilon}_n \sim \mathcal{N}(\boldsymbol{\mu}, \Sigma)$. We assume each event will last a number of time slots, and the objective is to continuously monitors $\mathbf{x}_n$ and detect events in a timely manner once they occur.

The event detection problem can cast into the quickest change detection (QCD) framework. However, compared with standard QCD formulations, there are two main difficulties that we need to address before applying classical QCD approaches to our problem: First, we need to have precise knowledge of $g(\cdot)$ so we can remove the temporal dependencies in the measurements in order to fit the i.i.d. assumption underlying the QCD framework; Second, we need to have distributional information about $\boldsymbol{\epsilon}_n$. In particular, we need to take the *unknown* mean shift $\boldsymbol{\mu}$ after event happens into account.

To address the first challenge, we propose to leverage the expressive power of the LSTM model to approximate $g(\cdot)$. The LSTM model can be trained in an offline fashion. Once trained, it should be able to closely track the temporal evolution of $\mathbf{x}_n$ when the system is stable. For the second challenge, we leverage the generalized likelihood ratio test (GLRT) to estimate the mean shift $\boldsymbol{\mu}$ *online*, and incorporate

it to the QCD framework.

In the following, we will first introduce the LSTM model and the QCD framework, and then elaborate how to utilize them for quick event detection in powder systems.

2.3 LSTM-based RNN for Forecasting

2.3.1 Recurrent Neural Networks (RNNs)

Typical neural network structures, such as Multilayer Perceptron (MLP), Deep Neural Network (DNN), highly depend on the assumption of independent observation data. However, in time series data, the observations are temporally correlated. It thus requires sophisticated models to take the temporal dependency into consideration when predicting time series data.

Different from the classical neural network structures, RNNs are designed to deal with time series, which allow temporal information to be passed from one time step to the next time step. As shown in Fig. 2.1, a loop structure is applied in RNNs for temporal information connection, thus making it suitable to model the time patterns among the data along the time axis.

Under vanilla RNN structure, with input time series $\{x_{t_1}, x_{t_2}, \cdots, \}$, the RNN calculates the hidden state h_t at each time slot recursively, and calculates the

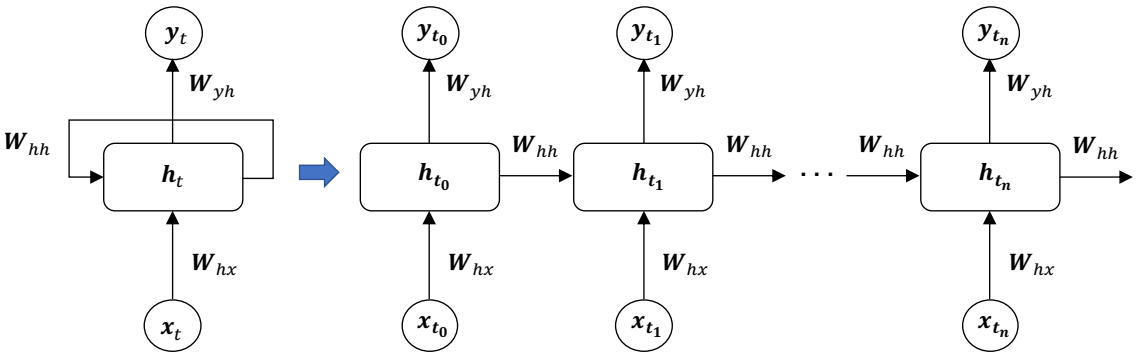


Figure 2.1: RNN structure.

output y_t with the current hidden state. Specifically,

$$\begin{aligned} h_{t_n} &= f(W_{hx}x_{t_n} + W_{hh}h_{t_{n-1}}), \\ y_{t_n} &= g(W_{yh}h_{t_n}), \end{aligned} \tag{2.2}$$

where W_{hx}, W_{hh}, and W_{hy} are the input-to-hidden weights, the hidden-to-hidden weights, and hidden-to-output weights, respectively. $f(\cdot)$ and $g(\cdot)$ are activation functions for the hidden layer and the output layer, respectively.

However, the vanilla RNN has a well known drawback called the gradient vanishing problem [42]. This is partially because the information flowing through neural networks passes through many stages of multiplication. Thus, the influence of the input for forecasting decays and vanishes over time. Therefore, vanilla RNN are not suitable for forecasting with long-term temporal dependencies.

2.3.2 Long Short Term Memory (LSTM) based RNN

To overcome the gradient vanishing problem, LSTM-based RNN was proposed [43]. Long Short Term Memory (LSTM) model is a recurrent neural network (RNN) first proposed in [43] to overcome the gradient vanishing problem in vanilla RNN [42]. The main idea is to import a memory cell into the RNN structure, which enables the storage and access of information over long periods of time. The memory cell runs straight down the entire chain, and LSTM uses three gates to optionally choose information to add into the memory cell. Such a structure can efficiently overcome the vanishing gradient problem, thus is suitable for time series forecasting problem with long term dependencies.

An LSTM cell is composed of a cell state, an input layer, an input gate, a forget gate and an output gate, as shown in Fig. 2.2. The historical information is stored in the cell state $\mathbf{c}_n$. The input gate decides which part of the input is worth storing. Such decision is made by considering current input $\mathbf{x}_n$ and previous output $\mathbf{h}_{n-1}$ together as follows: $\mathbf{i}_n = \sigma(\mathbf{W}_i \cdot [\mathbf{h}_{n-1}, \mathbf{x}_n] + \mathbf{b}_i)$, where $\mathbf{W}_i$ and $\mathbf{b}_i$ are the weight matrix and the bias vector of the input gate, respectively, $\sigma()$ is the sigmoid function with output range $(0, 1)$. When $\mathbf{i}_n$ is close to one, it tends to store the information in the cell state. On the other hand, when $\mathbf{i}_n$ is close to zero, it tends to drop the input information. The information selection is then applied by multiplying $\mathbf{i}_n$ with the input vector $\widetilde{\mathbf{c}}_n := \tanh(\mathbf{W}_c \cdot [\mathbf{h}_{n-1}, \mathbf{x}_n] + \mathbf{b}_c)$, where $\mathbf{W}_c$ and $\mathbf{b}_c$ are weight matrix and bias vector of the input layer, respectively.

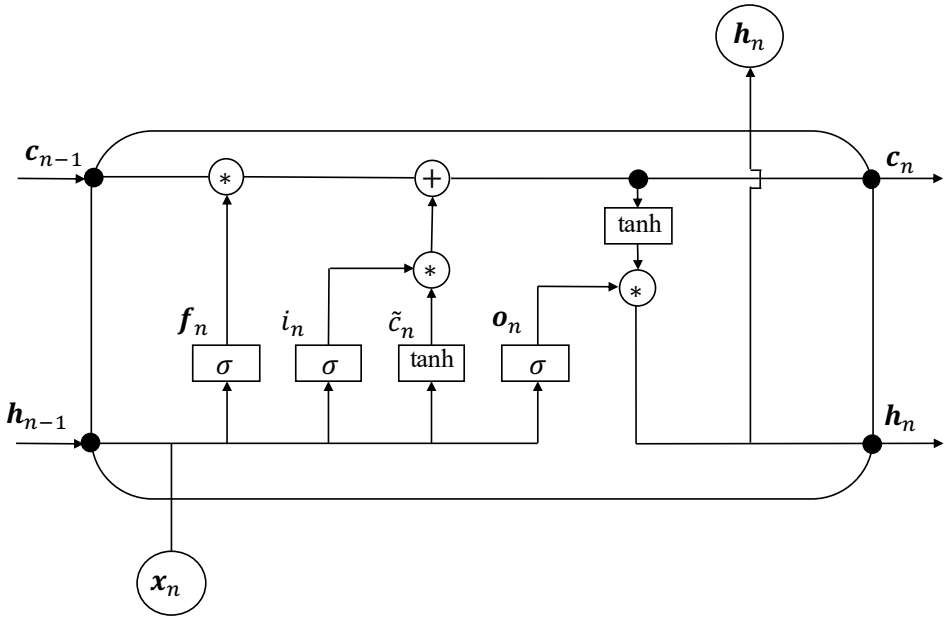


Figure 2.2: An LSTM cell.

The selected information is denoted as $\mathbf{c}_n^i := \mathbf{i}_n \odot \widetilde{\mathbf{c}}_n$, where $\odot$ is the element-wise multiplication, $\widetilde{\mathbf{c}}_n$ is the input information that combined by current input and previous output, and $\mathbf{c}_n^i$ is the selected information of the input layer that will be stored in the cell state.

Similar to the input gate, the forget gate decides which part of the information stored in the cell state should be forgotten. Specifically,

$$\begin{aligned} \mathbf{f}_n &= \sigma(\mathbf{W}_f \cdot [\mathbf{h}_{n-1}, \mathbf{x}_n] + \mathbf{b}_f), \\ \mathbf{c}_n^f &= \mathbf{f}_n \odot \mathbf{c}_{n-1}, \end{aligned} \tag{2.3}$$

where $\mathbf{W}_f$ and $\mathbf{b}_f$ are weight matrix and bias vector of the forget gate, respectively. Then, the new cell state is $\mathbf{c}_n = \mathbf{c}_n^i + \mathbf{c}_n^f$.

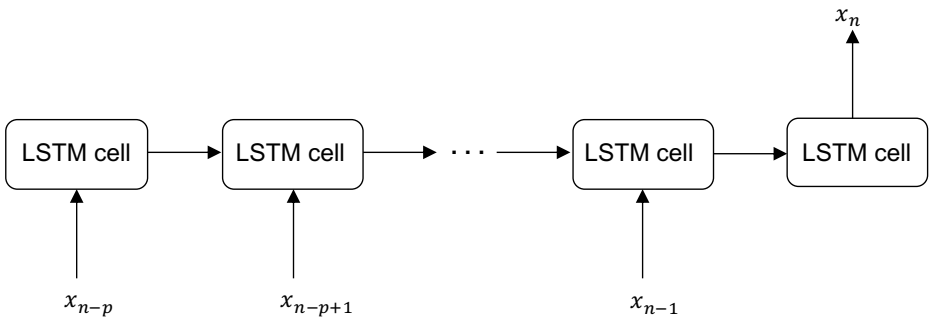


Figure 2.3: The LSTM model.

The output gate decides which part of information should be output, i.e.,

$$\begin{aligned} \mathbf{o}_n &= \sigma(\mathbf{W}_o \cdot [\mathbf{h}_{n-1}, \mathbf{x}_n] + \mathbf{b}_o), \\ \mathbf{h}_n &= \mathbf{o}_n \odot \tanh(\mathbf{c}_n), \end{aligned} \tag{2.4}$$

where $\mathbf{W}_o$ and $\mathbf{b}_o$ are weight matrix and bias vector of the forget gate, respectively.

The LSTM model consists of a chain of identical LSTM cells. The input of each LSTM cell corresponds to the measurements of a time step. Thus, the input layer can accommodate historical measurements $\{\mathbf{x}_{n-p}, \cdots, \mathbf{x}_{n-1}\}$ and the future value $\mathbf{x}_n$ can be predicted as shown in Fig. 2.3.

2.4 The Quickest Change Detection Framework

Consider a time series of measurements, denoted as $\{\mathbf{x}_1, \mathbf{x}_2, \cdots, \}$. Under the standard QCD framework, it assumes that there exits an unknown change point θ,

such that

$$\mathbf{x}_n \sim \begin{cases} f_0(\mathbf{x}_n), & \text{if } n < \theta, \\ f_1(\mathbf{x}_n), & \text{if } n \geq \theta, \end{cases} \tag{2.5}$$

where $f_0(\cdot)$ and $f_1(\cdot)$ are two different probability distribution functions. The objective of QCD is to obtain a detection procedure δ that maps the observation sequence $\{\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_n\}$ to an positive integer $\hat{\theta}$, i.e.,

$$\delta : \{\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_n\} \rightarrow \{\hat{\theta} : \hat{\theta} \leq n\}, \tag{2.6}$$

under certain constraints on metrics such as the probability of false alarm (PFA) and the average detection delay (ADD).

Denote the PFA and ADD under detection procedure δ as

$$\text{PFA}(\delta) := \mathbb{P}[\hat{t} < \theta], \quad \text{ADD}(\delta) := \mathbb{E}[\hat{t} - \theta | \hat{t} \geq \theta], \tag{2.7}$$

where $\hat{t}$ is the time when the detector declares a change and stops to take more measurements. Then, the change detection problem is formally formulated as follows:

$$\min_{\delta} \quad \text{ADD}(\delta), \text{ s.t. } \text{PFA}(\delta) \leq \alpha. \tag{2.8}$$

A typical change point detection method is to monitor the logarithm of the likelihood ratio $f_1(\mathbf{x}_n)/f_0(\mathbf{x}_n)$. A change point can be detected if the two distributions are significantly different. Classical approaches can be classified into the Bayesian framework and the non-Bayesian framework. In the Bayesian framework, it assumes that the change point is a random variable with known prior distribution, and one of the well known approach is Shiryaev procedure [44]. In the non-Bayesian framework, it does not assume any distribution on the change point. Classical approaches include the Cumulative Sum (CUSUM) procedure and its variants.

Since it is hard or even impossible to know when an event will happen beforehand in power systems, in this work, we focus on non-Bayesian approaches, especially CUSUM based approaches.

The CUSUM procedure was first proposed in [45]. It can be expressed as $\delta_c(h) = \inf\{n : S_n \geq h\}$, where S_n is the cumulative statistic at time n, and h is a pre-defined threshold h. Different threshold h will lead to different tradeoff between PFA and ADD.

Denote L_n as the log likelihood ratio based on the observation at time n, i.e., $L_n = \log \frac{f_1(\mathbf{x}_n)}{f_0(\mathbf{x}_n)}$. Then, S_n can be expressed as

$$S_n = \max_{1 \leq k \leq n} \sum_{i=k}^{n} L_i. \tag{2.9}$$

It has a convenient recursion as follows

$$S_n = \max(0, S_{n-1}) + L_n, \quad S_0 = 0. \tag{2.10}$$

It was shown in [46] that CUSUM is asymptotically optimal, i.e., for a given tolerance level on PFA, CUSUM minimizes the worst case detection delay when the average run length goes to infinity.

2.5 LSTM-based Event Detection

We note that one critical assumption in the QCD framework is the i.i.d. assumption on the measurements $\mathbf{x}_n$ before and after the change point. However, in power systems, the raw measurements $\mathbf{x}_n$ are in general not i.i.d. in time. In order to fit the QCD framework, in the following, we will first use an LSTM model to approximate the function $g(\cdot)$ in (2.1) and predict $\mathbf{x}_n$ based on past measurements.

Given the expressive power of LSTM, we expect that the output of LSTM, denoted as $\hat{\mathbf{x}}_n$, will be very close to $g(\mathbf{x}_{n-p}, \ldots, \mathbf{x}_{n-1})$. Thus, the prediction error, denoted as $\mathbf{x}_n - \hat{\mathbf{x}}_n$, will be very close to $\mathbf{e}_n$. Therefore, by utilizing the statistics of the prediction error for detection, the i.i.d. assumption required by the QCD framework can be satisfied.

Specifically, in the following, we will assume $\mathbf{e}_n = \mathbf{x}_n - \hat{\mathbf{x}}_n$ in the following.

Then, if an event happens at time θ, we have

$$\mathbf{e}_n \sim \begin{cases} \mathcal{N}(\mathbf{0}, \Sigma), & \text{if } n < \theta, n > \theta + \tau, \\ \mathcal{N}(\boldsymbol{\mu}, \Sigma), & \text{if } n \in [\theta, \theta + \tau], \end{cases} \tag{2.11}$$

where τ is the duration of the event. We assume τ is sufficiently long so an event can be detected before it ends.

Our objective is to detect the change point as quickly as possible based on the prediction error. We note that Σ can be learned from training data, thus is assumed to be known, while $\boldsymbol{\mu}$ is unknown due to the unforeseen events.

A well-known approach to handle unknown parameters in QCD is to incorporate the generalized likelihood ratio test (GLRT) to the CUSUM procedure [46]. The main idea is to use all the past observations to estimate the unknown parameters with the maximum likelihood estimator at each time step. The CUSUM statistic is then calculated using the estimated parameters. Finally, the GLRT-CUSUM is given by maximizing the log-likelihood ratio test statistic over all possible change time k as follows:

$$S_n = \max_{1 \le k \le n} \sum_{i=k}^{n} \log \frac{f_1(\mathbf{e}_n | \hat{\boldsymbol{\mu}}_k)}{f_0(\mathbf{e}_n)}, \tag{2.12}$$

where $\hat{\boldsymbol{\mu}}_k$ is the maximum likelihood estimate of $\boldsymbol{\mu}$ assuming a change point happens at time k.

One drawback of GLRT-CUSUM statistic is, it does not have the convenient

recursion as CUSUM statistic, thus having a heavy computational complexity. Since it requires to consider all previous observations and all potential parameters at each time step, it may not be very practical to apply it in real-time quickest change detection.

To avoid the complicated maximum likelihood estimation, Rao test is adopted to simplify the calculation [47]. Rao test is asymptotically equivalent to GLRT but has a much simpler calculation by taking derivative with respect to the unknown parameters. Under our setting, Rao test statistic can be expressed as follows:

$$\mathcal{R}(\mathbf{e}_n) = \frac{\partial L_n}{\partial \boldsymbol{\mu}}\bigg|^{T}_{\boldsymbol{\mu}=\mathbf{0}} \cdot [\mathbf{J}^{-1}(\boldsymbol{\mu})|_{\boldsymbol{\mu}=\mathbf{0}}] \frac{\partial L_n}{\partial \boldsymbol{\mu}}\bigg|_{\boldsymbol{\mu}=\mathbf{0}} \tag{2.13}$$

where $\mathbf{J}(\boldsymbol{\mu})$ is the Fisher information matrix, defined as $\mathbf{J}(\boldsymbol{\mu}) = -\frac{\partial^2 L_n}{\partial \boldsymbol{\mu}^2}$. Then, (2.12) can be reduced to

$$\bar{S}_n = \max_{1\le k\le n} \sum_{i=k}^{n} \mathcal{R}(\mathbf{e}_i), \tag{2.14}$$

which is similar to the form of (2.9) and admits convenient recursion as follows

$$\bar{S}_n = \max(0, \bar{S}_{n-1}) + \mathcal{R}(\mathbf{e}_n), \quad \bar{S}_0 = 0. \tag{2.15}$$

With the recursive expression, we can track the approximate CUSUM statistic of $\mathbf{e}_n$ at each time step, and determine whether there is a change point or not. If

$\bar{S}_n$ reaches the pre-defined threshold h, then a change point is declared.

2.6 Numerical Results

We test the proposed prediction and detection procedure on a dataset collected by Oklahoma Gas & Electric Company (OG&E) [48]. The dataset contains PMU measurements sampled from 285 buses in a 2-minute window, during which a line outage happens. The measurements are collected at a rate of 30 data points per second. In order to illustrate our main idea, in the following, we only use the change of frequency (df/dt) as the variable for the event detection. We point out that both the LSTM model and the CUSUM based detection procedure can accommodate multiple variables. Thus, other measured quantities, such as voltage and current, can be exploited together to improve the detection performance potentially.

Under the LSTM model, we use 50 consecutive df/dt measurements to predict the df/dt at the next time step. The model parameters are listed in Table 2.1. We can see that the predicted value using the LSTM model matches the measurement well.

Table 2.1: Parameters of the LSTM model.

Time steps	Hidden layer size	Optimizer	Learning rate	Dropout	Epochs
50	100	Adam	1.2e-3	0.1	120

Then, we plot the histogram of the training error, prediction error before change point and prediction error after change point to verify the normal distribution

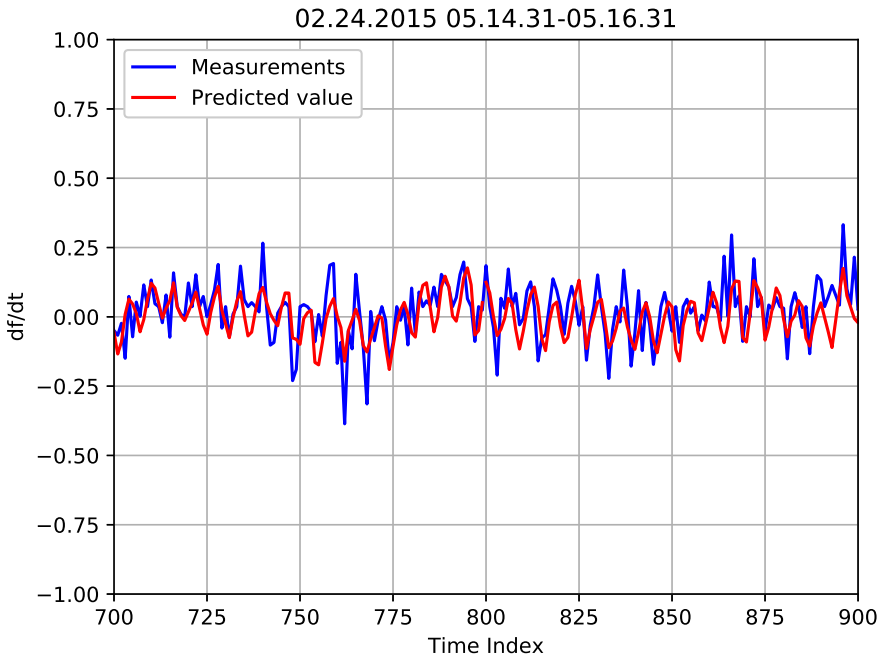


Figure 2.4: df/dt measurements and predictions at 'Sooner-Cleveland'.

assumption.

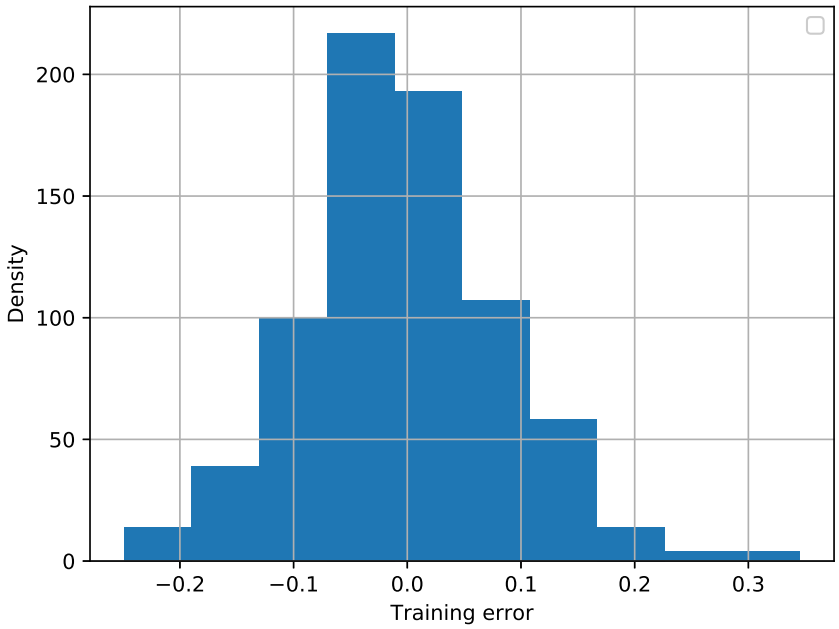


Figure 2.5: Histogram of training error.

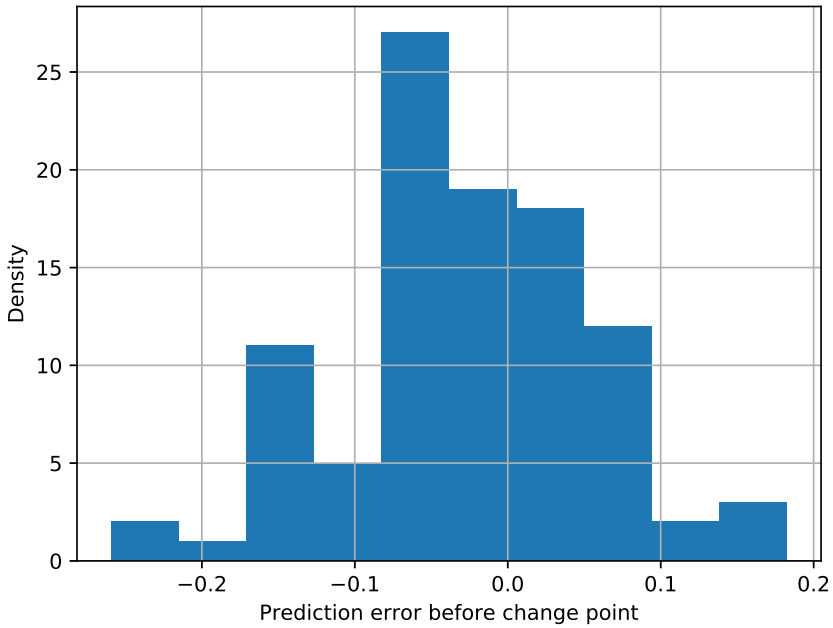


Figure 2.6: Histogram of prediction error before the change point.

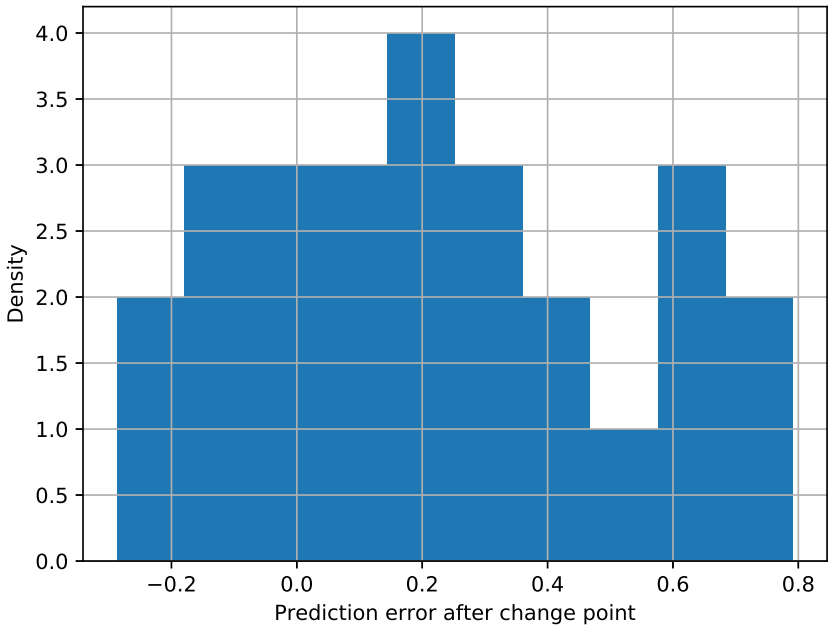


Figure 2.7: Histogram of prediction error after the change point.

Fig. 2.5 and Fig. 2.6 show that the training error and prediction error before the change point follow the same normal distribution, which implies that the parameters can be learned during the training process and can be assumed as known. Fig. 2.7 shows that the distribution of the prediction errors after the change point has a shift in mean, as the LSTM predictor cannot predict the df/dt value accurately when the system enters an unstable state after the anomalous event. However, the distribution still can be approximated as a normal distribution with the same variance, which verifies our assumption.

Next, we test the event detection performance on all 285 bus. We train an LSTM model for each bus and use the prediction error for event detection under different

threshold h. For each threshold h, we record PFA as the percentage of buses experiencing false alarm and record ADD as the average detection delay over all buses. For comparison, we also evaluate the performance of two baseline algorithms. For the first baseline algorithm, we use a linear regression (LR) model for prediction and follow the same CUSUM detection procedure as in our proposed approach. For the second baseline algorithm, instead of adopting CUSUM, we simply use the accumulated prediction error from the LSTM model for event detection. Specifically, we monitor the absolute value of prediction error and compare it with a predefined threshold th_1, which is a scaled standard deviation of the prediction errors during training. Another threshold th_2 is set to declare an event when the sum of the prediction errors that exceed th_1 reaches th_2. Fig. 2.10 shows the trade-off between PFA and ADD under those algorithms. As we observe, the proposed algorithm outperforms the baseline algorithms, especially when the PFA is low, which is a reasonable setting in practical systems. This indicates that the proposed algorithm can detect events more quickly.

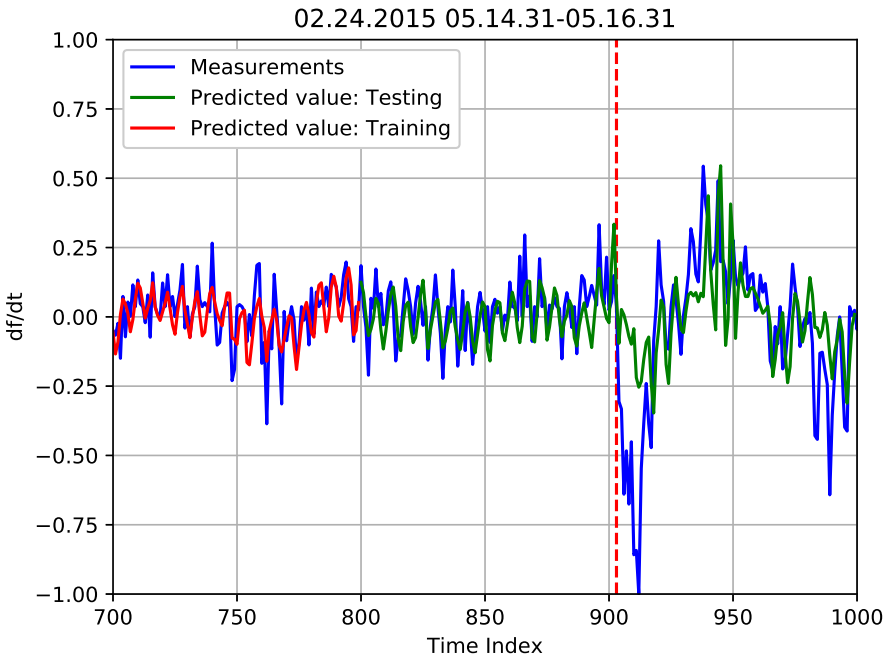


Figure 2.8: df/dt prediction.

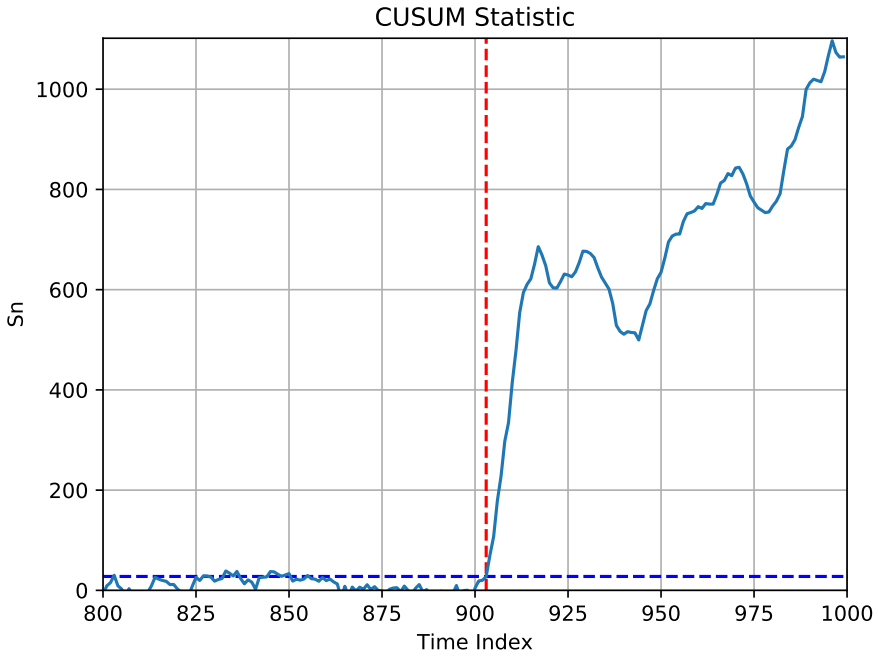


Figure 2.9: CUSUM Statistic.

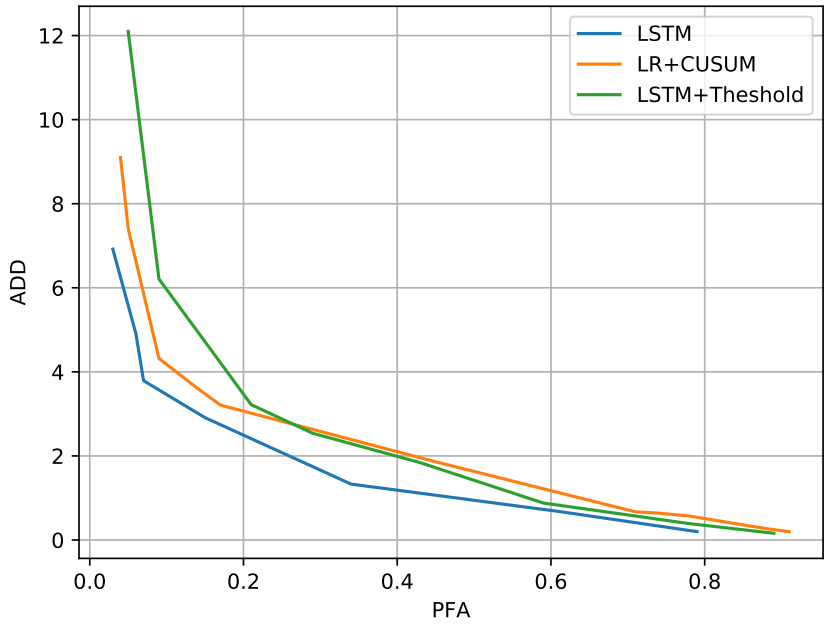


Figure 2.10: ADD versus PFA.

2.7 Conclusions

In this chapter, we proposed a data-driven approach to detect events in power systems in real time. An LSTM model was adopted to capture the temporal dependencies in the time series measurements. A CUSUM based procedure was then developed by tracking the statistics of the prediction errors. Numerical results based on a real-world PMU dataset were obtained to validate the superior performance of the proposed approach. Future work includes aggregating measurements from multiple buses to detect and locate events, as well as joint training of the LSTM models across buses.

Chapter 3

Optimal Electric Vehicle Charging Scheduling with Time-varying Profits

Electric vehicle charging scheduling to maximize the revenue of charging station while satisfying customer demands is challenging, mainly due to the time-varying nature of user demands and electricity price. In this paper, we develop an admission control and scheduling mechanism to jointly consider the revenue of charging stations and the service requirements of customers. We first consider an offline setting. Given a set of inflexible service demands, our objective is to design an admission control and scheduling scheme, so that the admitted demands can be fulfilled while the revenue of the charging station is maximized. We first propose a calculus-based scheduling algorithm, and show that for a given feasible demand set, it maximizes the revenue of the charging station. We then consider the general case where the charging station may not be able to fulfill all demands and admission control is needed. We first prove that the general case is NP-complete, and then develop a

heuristic algorithm to greedily decline a subset of demands until the remaining demands can be fulfilled. We then consider an online setting where the demand information is available causally, and develop an online admission control and scheduling algorithm. Simulation results indicates that the proposed algorithms outperform baseline algorithms.

3.1 Introduction

The recent few years have witnessed a resurgence of electric vehicles (EV). Compared with conventional gas-powered vehicles, EV exhibits various advantages, such as increased energy security, lower fuel costs, and reduced carbon emissions [49]. The ever increasing adoption of EVs requires the charging stations to coordinate the dynamic charging demands in a sophisticated manner. The objective is often to maintain the stability of the power system [50], as well as to reduce the energy costs and accommodate more service demands. What makes the problem complicated is the time-varying nature of the electricity price and the charging demand arrivals.

Various EV charging scheduling algorithms have been proposed to optimize different performance metrics. From the power grid's perspective, [51] uses quadratic and dynamic programing techniques to minimize the power losses and to maximize the main grid load factor. The objective in [52] is to minimize the distribution system load variance. References [53] and [54] consider the valley-filling problem by scheduling the charging of EVs to flatten the load profile. Reference [43] jointly

considers the load profile flattening and charging cost reduction. In [55], factors that affect battery life are quantified and their impact on battery degradation and power grid support is analyzed.

Other works have focused on benefiting either the charging station side, such as maximizing its revenue [56–61], or the EV users' side, such as to guarantee the qualityof service [62–66]. In [57], the optimal scheduling is analyzed from an economic perspective. The objective is to minimize the charge station's cost with budget constraints in an unregulated electricity market. In [59], an optimal charging power allocation is studied to minimize the charging fees. Reference [60] aims to maximize the average revenue by considering the charging scheduling problem as a constrained binary programming problem. In [61], an online charging algorithm is proposed to minimize the cost without future information, where the electricity price is modeled as a linear function of the instant load. In [62], a pre-commitment algorithm is proposed to ensure the pre-committed demands are fulfilled before their departures. Reference [65] investigates the optimal charging scheduling in order to minimize the charging waiting time.

Beside those works that mainly focus on a single optimization objective, there also exist works that jointly consider satisfying users' service requirements and maximizing charging stations' revenue. In [67], a two-stage process including admission control and charging scheduling is proposed to balance the trade-off between admitting more charging demands and failing to meet their deadlines.

Reference [68] considers a deadline restricted utility maximization problem where the utility generated by charging an EV in a time slot depends on the accumulated served demand of that EV. A heuristic admission control and utility based scheduling algorithm is proposed to benefit both the charging station and the customers. In [69], a two-stage approximate dynamic programming framework is proposed to determine the optimal charging strategy, with the objective of minimizing the energy cost and satisfying the charging demands. Reference [70] schedules the charging of EVs based on their laxity and remaining processing time to minimize the cost and uncompleted charging demands. Most of these works add a penalty for the unsatisfied demands, and cannot guarantee that the charging demands are fully satisfied.

This problem is also related to the deadline scheduling problem [71–73]. In [71], an online scheduling algorithm is proposed to maximize the profit of a single processor, where a penalty is imposed if an admitted task is not completed by its deadline. In [72], admission control and scheduling is considered for a mixture of local renewable energy sources and power grids. In [73], a network flow deadline scheduling problem is considered. However, such framework cannot accommodate the time-varying profit rate setting in EV charging.

In this chapter, we consider the coordination of multiple charging demands at a charging station with time-varying profits. We assume the energy demands of the EVs are *inflexible*, i.e., once a demand is admitted, the charging station

should guarantee that the demanded amount is served before its deadline. Our objective is to develop an admission control and charging scheduling mechanism to maximize the revenue of the charging station. We first propose a novel calculus based approach to optimize the overall revenue for a given admitted feasible demand set. We then study the general case with admission control, and show that the problem is NP-complete. Next, we propose a heuristic admission control algorithm for the general overloaded case. Finally, we consider an online setting and propose an online algorithm. The algorithms are evaluated through simulations.

3.2 Problem Formulation

We consider a plug-in EV charging station, where EVs come and request certain amounts of energy to be charged before a deadline. Once a demand is received, the control center will decide whether to admit it or decline it. After a demand is admitted, the control center will ensure that the requested energy amount is fulfilled before the deadline.

Formerly, we consider a discrete-time model where the time duration of interest is divided into N time slots (TSs), indexed by $1, 2, \ldots, N$. The duration of each TS is τ hours. Each charging demand f_i is represented by a three-tuple $< s_i, e_i, d_i >$, where e_i is the requested energy amount (in kWh), s_i and d_i are the arrival TS and the deadline TS, respectively. We assume a demand arrives at the beginning of s_i and must be completed before the end of d_i. For each demand, its deadline must be

greater than its arrival time, i.e., $1 \leq s_i \leq d_i \leq N$. We consider an offline setting, where a set of demands $\mathcal{F} = \{f_1, f_2, \ldots, f_m\}$ arrive over the N TSs. Without loss of generality, we assume $d_1 \leq d_2 \leq \ldots \leq d_m$.

For a TS $n \in \{s_i, s_{i+1}, ..., d_i\}$, denote the amount of energy charged for demand f_i as x_n^i. Let x_n be the total charged energy at TS n, i.e., $x_n = \sum_{i:f_i \in \mathcal{F}(n)} x_n^i$, where $\mathcal{F}(n) := \{f_i \in \mathcal{F} : s_i \leq n \leq d_i\}$. A demand f_i is said to be fulfilled, if and only if the amount of charged energy by its deadline reaches its demanded energy e_i, i.e.,

$$e_i = \sum_{n=s_i}^{d_i} x_n^i. \tag{3.1}$$

Further, we assume that the maximum charging power the station can supply in any TS is p_{max}, i.e.,

$$x_n \leq p_{max} \cdot \tau, \quad n = 1, 2, ..., N. \tag{3.2}$$

We consider a time-varying electricity pricing model under which the cost per unit energy varies in time. We denote the *profit* per unit, i.e., the difference between the unit price it charges the customers and it pays the power company in TS n as R_n dollar/kWh. We assume $R_n \geq 0, \forall n$. Therefore, for a given set of admitted demands $\mathcal{F}_\mathcal{A}$, and their charging schedules, the profit the charging station will

obtain equals

$$P = \sum_{n=1}^{N} \sum_{i:f_i \in \mathcal{F}_{\mathcal{A}}(n)} x_n^i \cdot R_n. \tag{3.3}$$

We first focus on an offline setting. Then, for a given set of demands $\mathcal{F}$, our objective is to develop a mechanism $\{\mathcal{F}_{\mathcal{A}}, \mathcal{S}_{\mathcal{A}}\}$ to decide the admitted demands $\mathcal{F}_{\mathcal{A}}$, $\mathcal{F}_{\mathcal{A}} \subseteq \mathcal{F}$, as well as their charging schedule $\mathcal{S}_{\mathcal{A}} := \{\{x_n^i\}_{n \in f_i}\}_{f_i \in \mathcal{F}_{\mathcal{A}}}$, so that the revenue of the charging station is maximized, while all of the admitted demands are fulfilled. Mathematically, the optimization problem can be formulated as

$$\max_{\{\mathcal{F}_{\mathcal{A}}, \mathcal{S}_{\mathcal{A}}\}} \quad \sum_{n=1}^{N} \sum_{i:f_i \in \mathcal{F}_{\mathcal{A}}(n)} x_n^i \cdot R_n, \quad \text{s.t.} \quad (3.1) - (3.2). \tag{3.4}$$

In the following, we will first focus on revenue maximization for a given admitted feasible demand set $\mathcal{F}_{\mathcal{A}}$. We will then study a more general setting where the charging demands in $\mathcal{F}$ may not be satisfied simultaneously. We will show that the optimization problem is NP-complete, and then propose a low-complexity algorithm to solve it approximately.

3.3 Optimal Scheduling

In this section, we focus on the scheduling of a given admitted demand set $\mathcal{F}_{\mathcal{A}} = \{f_1, f_2, ..., f_m\}$ by assuming that there exists a schedule $\mathcal{S}_{\mathcal{A}}$ to fulfill all demands in

$\mathcal{F}_\mathcal{A}$ while satisfying the maximum charging power constraint. We have the following observation.

Lemma 1. *A scheduling $\mathcal{S}_\mathcal{A}$ with total charged energy $\{x_n\}_{n=1}^N$ is feasible, if and only if*

$$x_n \leq p_{max} \cdot \tau, \quad n \in \{1, 2, ..., N\}, \tag{3.5}$$

$$\sum_{n=1}^{N} x_n = \sum_{i=1}^{m} e_i, \tag{3.6}$$

$$\sum_{n=a}^{b} x_n \geq \sum_{i:a \leq s_i \leq d_i \leq b} e_i, \quad 1 \leq a \leq b \leq N, \tag{3.7}$$

$$\sum_{n=1}^{b} x_n \leq \sum_{i:s_i \leq b, f_i \in \mathcal{F}_\mathcal{A}} e_i, \quad b \in \{1, 2, ..., N\}. \tag{3.8}$$

Proof: First, we show that those conditions are *necessary* for the feasibility of the scheduling policy. If $\mathcal{S}_\mathcal{A}$ is feasible, equations (3.5) and (3.6) are satisfied due to constraints (3.1) and (3.2). Equation (3.7) comes from the fact that the total charged energy over any period should be greater than or equal to the total amount of energy requested by the demands whose arrival time and deadline time are both within this period; Equation (3.8) is due to the fact that the cumulative charged energy by a TS should be less than or equal to the total demanded energy by that TS.

Next, we prove that given any $\{x_n\}_{n=1}^N$ satisfying those conditions, we are able to design a charging schedule $\{x_n^i\}_{n,i}$ to ensure the demands in $\mathcal{F}_\mathcal{A}$ are satisfied, i.e., the conditions are *sufficient* for the feasibility of a schedule $\mathcal{S}_\mathcal{A}$. The policy is to allocate

$\{x_n\}$ to the demands in the order of their deadlines. Under this policy, the charging station will first serve f_1 with the maximum energy available start from s_1 to d_1 until f_1 is satisfied, i.e., it will let $x^1_{s_1} = \min(x_{s_1}, e_1)$, $x^1_{s_1+1} = \min(x_{s_1+1}, e_1 - x^1_{s_1})$, ..., $x^1_{d_1} = \min(x_{d_1}, e_1 - \sum_{n=s_1}^{d_1-1} x^1_n)$. According to the condition in (3.7), f_1 can be fully satisfied as $\sum_{n=s_1}^{d_1} x_n \geq e_1$. Next, we move on to demand f_2 with the remaining energy $\{\bar{x}_n\}$, where $\bar{x}_n = x_n - x^1_n$, for $n \in \{s_1, s_1+1, \ldots, d_1\}$, and $\bar{x}_n = x_n$ otherwise. Then, depending the arrival and deadline time of f_2, we have three possible cases:

Case 1: $s_2 > d_1$. For this case we have $\sum_{n=s_2}^{d_2} x_n \geq e_2$ according to (3.7).

Case 2: $s_1 \leq s_2 \leq d_1$. If $x^1_{s_2} = 0$, then $\bar{x}_{s_2} = x_{s_2}$, $\sum_{n=s_2}^{d_2} \bar{x}_n \geq e_2$ according to (3.7). If $x^1_{s_2} > 0$, we still have $\sum_{n=s_2}^{d_2} \bar{x}_n \geq e_2$ as $\sum_{n=s_1}^{d_2} x_n \geq e_1 + e_2$ and $\sum_{n=s_1}^{d_1} x^1_n = e_1$.

Case 3: $s_2 < s_1$. For this case, we have $\sum_{n=s_2}^{d_2} \bar{x}_t \geq e_2$ as $\sum_{n=s_2}^{d_2} x_n \geq e_1 + e_2$ and $\sum_{n=s_1}^{d_1} x^1_n = e_1$.

Thus, for all three cases, by allocating $\{\bar{x}_n\}_{n=s_2}^{d_2}$ sequentially to f_2 in a greedy fashion, f_2 can be fully satisfied.

Applying the same idea recursively, we can prove that $f_3, \ldots, f_m$ can be fully satisfied by the deadline based allocation policy, which ensures the sufficiency of the conditions for the feasibility of a scheduling. ■

Based on Lemma 1, with admitted feasible $\mathcal{F}_\mathcal{A}$, the optimization problem then

can be reduced to

$$\max_{\{x_n\}} \quad \sum_{n=1}^{N} x_n \cdot R_n \quad \text{s.t.} \quad (3.5) - (3.8). \tag{3.9}$$

Intuitively, to solve (3.9), we need to charge more energy in those TSs with higher profit rates to earn more profit. We will first determine $\{x_n\}_{n=1}^{N}$, the total charged energy in each TS using a calculus based approach. Then, in the second stage we will use the aforementioned deadline based allocation policy to allocate $\{x_n\}_{n=1}^{N}$ to the demands and decide $\{x_n^i\}$.

3.3.1 Calculus Based Scheduling

In order to facilitate the analysis, we first introduce a calculus-based scheduling as follows. We define a cumulative energy demand upper curve $A(t)$ and a minimum supply curve $D(t)$, $t \in [0, N\tau]$, based on the demand set $\mathcal{F}_{\mathcal{A}}$. Specifically, we have

$$A(t) = \sum_{i:(s_i-1)\tau<t} e_i, \quad D(t) = \sum_{i:d_i\tau\leq t} e_i. \tag{3.10}$$

As shown in Fig. 3.1(b), both of the curves are piecewise constant.

Next, we define a cumulative served demand function $S(t)$ over $[0, N\tau]$ as follows. For $t \in ((n-1)\tau, n\tau]$, $n = 1, 2 \ldots, N$,

$$S(t) = \sum_{j=1}^{n-1} x_j + x_n \frac{(t-(n-1)\tau)}{\tau}, \tag{3.11}$$

and $S(0) = 0$. Then, we have the following observations.

Lemma 2. *Under any feasible scheduling policy satisfying (3.5)-(3.8), $S(t)$ is a continuous, monotonically increasing, and piecewise linear curve lying between $A(t)$ and $D(t)$. Besides, the slope of $S(t)$ is always bounded by p_{max}, $S(N\tau) = \sum_{i:f_i \in \mathcal{F}_\mathcal{A}} e_i$, and for $1 \leq a \leq b \leq N$,*

$$S(b\tau) - S((a-1)\tau) \geq \sum_{i:a \leq s_i \leq d_i \leq b} e_i. \tag{3.12}$$

Lemma 2 essentially translates the conditions (3.5)-(3.8) into the properties of $S(t)$. Therefore, to optimize the profit, it is equivalent to find a curve $S(t)$ satisfying the conditions in Lemma 2 while ensuring that the *weighted* summation of the increments of $S(t)$ in individual TSs is maximized, where the weight corresponds to the profit rate in that TS. Since the total increment (i.e., $S(N\tau)$) is fixed, it thus becomes intuitive to greedily maximize the increments of $S(t)$ in the TSs with higher profit rates while abiding Lemma 2.

Motivated by this observation, we propose an algorithm to sequentially identify the increment of $S(t)$ in each TS until all demands are satisfied, as detailed in Algorithm 3. First, we sort the TS indices in the descending order of R_t, and denote the sorted indices of the TSs as $I_1, I_2, \ldots, I_N$. Then, starting from I_1, we identify a lower bound for $S((I_1 - 1)\tau)$ according to Algorithm 4, denoted as $\underline{S}((I_1 - 1)\tau)$, and an upper bound for $S(I_1\tau)$ based on Algorithm 5, denoted as

$\bar{S}(I_1\tau)$. We then let x_{I_1}, i.e., the increment of $S(t)$ in TS I_1, be the maximum between $\bar{S}(I_1\tau) - \underline{S}((I_1-1)\tau)$ and $p_{max}\tau$. With x_{I_1} fixed, we then move on to TS I_2 to identify x_{I_2}, based on the constraints in Lemma 2 as well as the determined x_{I_1}. We repeat the process until all demands are satisfied or all TS has been examined.

Algorithm 3 Calculus-based Scheduling

Initialize: $k = 1$, $x_n = 0$, $n = 1, \ldots, N$.
Sort $\{R_n\}$ s.t. $R_{I_1} \geq R_{I_2} \geq \ldots \geq R_{I_N}$;
while $\sum x_n < \sum e_i$ or $k > N$ **do**
    $\underline{S}(I_k - 1)\tau) \leftarrow$ LOWERBOUND$(\mathcal{F}_{\mathcal{A}}, \{x_n\}, I_k)$;
    $\bar{S}(I_k\tau) \leftarrow$ UPPERBOUND$(\mathcal{F}_{\mathcal{A}}, \{x_n\}, I_k)$;
    $x_{I_k} = \min\{p_{max} \cdot \tau, \bar{S}(I_k\tau) - \underline{S}((I_k - 1)\tau)\}$;
    $k = k + 1$;
end while
Return $\{x_n\}$.

Algorithm 4 LowerBound$(\mathcal{F}_{\mathcal{A}}, \{x_n\}, p)$

$\{x_n^i\} \leftarrow \text{DBA}(\{f_n : d_n < p\}, \{x_n\}_{n=1}^{p-1})$;
$\underline{S}((p-1)\tau) = D((p-1)\tau) + \sum_{n=1}^{p-1} x_n - \sum_{n=1}^{p-1} \sum_{i:d_i<p} x_n^i$;
Return $\underline{S}((p-1)\tau)$;

Algorithm 5 UpperBound$(\mathcal{F}_{\mathcal{A}}, \{x_n\}, p)$

$\{x_n^i\} \leftarrow \text{DBA}(\{f_n : s_n > p\}, \{x_n\}_{n=p+1}^{N})$;
$\bar{S}(p\tau) = A(p\tau) - \sum_{n=p+1}^{N} x_n + \sum_{n=p+1}^{N} \sum_{i:s_i>p} x_n^i$;
Return $\bar{S}(p\tau)$;

3.3.2 A Special Case: First-in-first-out Demands

In order to illustrate the process of identifying the lower and upper bounds $\underline{S}((n-1)\tau)$ and $\bar{S}(n\tau)$, we start with a special case, where all of the demands are first-

Algorithm 6 DBA($\{f_i\}_{i\in\mathcal{I}}, \{x_n\}_{n=a}^{b}$)

Let available energy $\bar{x}_n = x_n$, $n \in [a, b]$;
while $\mathcal{I} \neq \emptyset$ **do**
    $i = \min\{j : f_j \in \mathcal{I}\}$
    for $n = s_i, s_i + 1, \ldots, d_i$ **do**
        Allocate $\bar{x}_n$ to f_i until it is satisfied;
        Update available energy $\bar{x}_n$; $\mathcal{I} = \mathcal{I}\backslash\{i\}$;
    end for
end while
Return $\{x_n^i\}$;

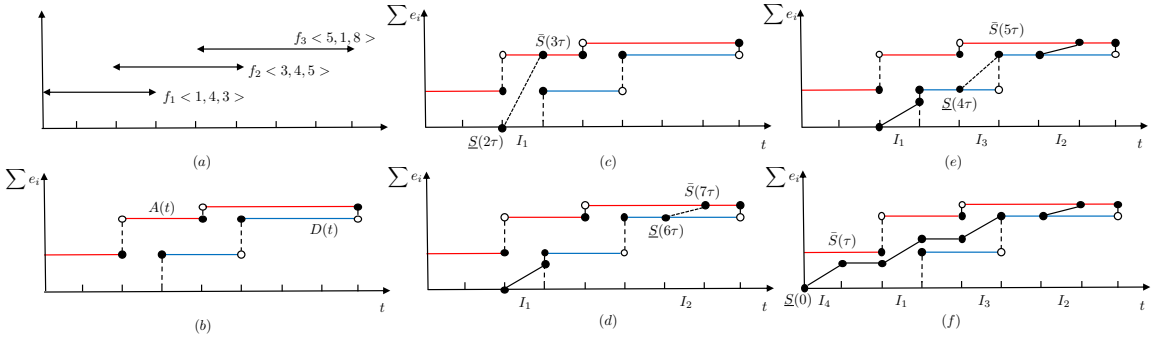


Figure 3.1: An illustration for the FIFO case.

in-first-out (FIFO), i.e., if $s_i \leq s_j$, we must have $d_i \leq d_j$ as well. We have the following observations.

Lemma 3. *When all of the demands are FIFO, the condition in (3.7) is equivalent to:*

$$\sum_{n=1}^{b} x_n \geq \sum_{i:1\leq s_i\leq d_i\leq b} e_i, \quad 1 \leq b \leq N. \tag{3.13}$$

Proof: To show the equivalence, it suffices to show that equation (3.13) implies

(3.7). We note that for any $a \leq b$, we have

$$\sum_{n=1}^{a-1} x_n \leq \sum_{i:s_i \leq a-1} e_i \tag{3.14}$$

based on (3.8). Thus, combining (3.13) and (3.14), we have

$$\sum_{n=a}^{b} x_n \geq \sum_{i:1 \leq s_i \leq d_i \leq b} e_i - \sum_{i:s_i \leq a-1} e_i. \tag{3.15}$$

In the FIFO case, if there exists a demand with $s_i \leq a-1$ and $d_i > b$, then $\sum_{i:a \leq s_i \leq d_i \leq b} e_i = 0$, and (3.7) becomes a redundant constraint; If there does not exist any demand with $s_i \leq a-1$ and $d_i > b$,

$$\sum_{i:1 \leq s_i \leq d_i \leq b} e_i - \sum_{i:s_i \leq a-1} e_i = \sum_{i:a \leq s_i \leq d_i \leq b} e_i. \tag{3.16}$$

Thus (3.13) is identical to (3.7). ■

The condition in (3.13) means that $S(t) \geq D(t), \forall t$. Therefore, for this special case, the upper and lower bounds of $S(t)$ are completely determined by $D(t)$ and $A(t)$.

Fig. 3.1 is an example illustrating the procedure of identifying the lower and upper bounds of $S(t)$. We start with $x_n = 0, \forall n$. At TS $I_1 = 3$, the corresponding lower and upper bounds are $\underline{S}(2\tau) = D(2\tau)$ and $\bar{S}(3\tau) = A(3\tau)$, respectively. We let $x_3 = \min\{p_{max} \cdot \tau, \bar{S}(3\tau) - \underline{S}(2\tau)\} = p_{max} \cdot \tau$. For TS $I_2 = 7$, the upper and

lower bounds are $D(6\tau)$ and $A(7\tau)$, respectively, as x_3 does not affect the lower bound $\underline{S}(6\tau)$. Then, we consider TS $I_3 = 5$. We have $\underline{S}(4\tau) = D(4\tau)$ as x_5 does not affect the lower bound $\underline{S}((4\tau)$, while $\bar{S}(5\tau) = A(8\tau) - x_7$, as x_7 has been decided. For TS $I_4 = 1$, the lower bound is $D(0)$, and the upper bound is affected by x_3, x_5 and x_7, as shown in the Fig. 3.1(f). The algorithm stops after x_1 is determined, as $\sum_{n=1}^{N} x_n = \sum_{i=1}^{3} e_i$.

In general, for FIFO demands, after $\{x_n\}$ has been determined in $I_1, \cdots, I_{k-1}$, in TS $I_k = n$, the lower bound of $S((n-1)\tau)$ can be recursively calculated as

$$\underline{S}((n-1)\tau) = \max\{D((n-1)\tau), x_{n-1} + \underline{S}((n-2)\tau)\}, \tag{3.17}$$

where $\underline{S}(0) = D(0)$. Similarly, the upper bound of $S(n\tau)$ can be recursively determined as

$$\bar{S}(n\tau) = \min\left(A(n\tau), \bar{S}((n+1)\tau) - x_{n+1}\right), \tag{3.18}$$

where $\bar{S}(N\tau) = A(N\tau)$.

3.3.3 Demands with General Arrival Times and Deadlines

For the general case where the demands may not be FIFO, equation (3.7) is not equivalent to (3.13) anymore. Thus, lying in between $A(t)$ and $D(t)$ cannot guarantee that $S(t)$ is always feasible, as the constraint in (3.7) may not be satisfied.

We use the example in Fig. 3.2 to illustrate it. In this example, we have $s_3 < s_1, s_2$. Thus x_1 can not be used to charge f_1 and f_2. Therefore, having $S(3\tau)$ above $D(3\tau)$ cannot guarantee that f_1 is fully satisfied. If we have $x_1 = 3$, although $x_2 = 1$ and $x_3 = 0$, the lower bound $S(3\tau)$ should be $D(3\tau) + x_1$, as we should reserve $x_2 + x_3 \geq 3$ in order to make sure constraint (3.7) is satisfied when the charging scheduling is finalized.

In general, to find $\underline{S}((q-1)\tau)$ without violating constraint (3.7), we divide $S(t)$ into two parts: $D(t)$ and $\tilde{S}(t)$, i.e., $S(t) = D(t) + \tilde{S}(t)$. $D(t)$ corresponds to the cumulative energy that is charged for the demands that end before or at t, while $\tilde{S}(t)$ is the cumulative energy that is charged for the demands that are still active after t. Thus, to identify the lower bound on $S((q-1)\tau)$ with given $\{x_n\}_{n=1}^{q-1}$, we need to allocate $\{x_n\}_{n=1}^{q-1}$ to the demands that end before TS q as much as possible, so that the remaining energy that will be allocated to demands active in TS q is minimized. A deadline based allocation (DBA) algorithm is presented in Algorithm 6. With given energies $\{x_n\}$, Algorithm 6 allocates them to the demands in the order of their deadlines, until all energy is allocated, or all demands are satisfied. We will show that this algorithm maximizes the total energy allocated to the specified subset of demands in Lemma 4. Thus, the remaining energy is minimized. The upper bound $\bar{S}(q\tau)$ can be obtained using a similar idea.

Lemma 4. *Given a subset of demands $\{f_i\}_{i\in\mathcal{I}}$, and fixed total charging amounts $\{x_n\}_{n=a}^{b}$, Algorithm 6 solves the following optimization problem:*

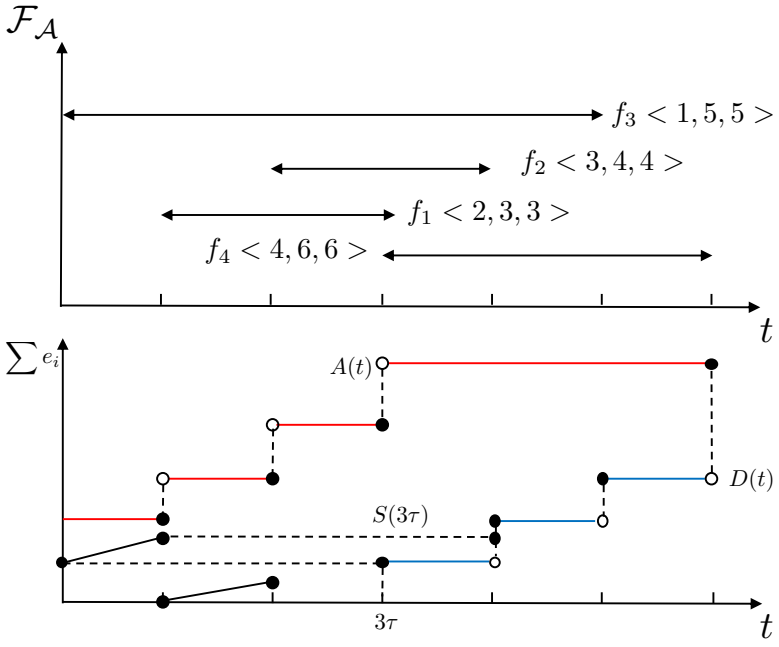


Figure 3.2: An example of general case.

$$\max \quad \sum_{i\in\mathcal{I}} \sum_{n=a}^{b} x_n^i, \tag{3.19}$$

$$\text{s.t.} \quad \sum_{n=s_i}^{d_i} x_n^i \le e_i, i \in \mathcal{I}, \sum_{i:f_i\in\mathcal{I}(n)} x_n^i \le x_n, a \le n \le b.$$

Proof: Assume $\{\hat{x}_n^i\}$ is the optimal solution to (3.19), and $\{x_n^i\}$ is the allocation given by Algorithm 6. We aim to show that $\{\hat{x}_n^i\}$ can be transformed to $\{x_n^i\}$ without compromising its optimality. With a slight abuse of notation, we index the flows in $\mathcal{I}$ as $1, 2, \cdots, |\mathcal{I}|$ in the order of their deadlines. Let $\hat{\mathcal{I}}(n)$ be the set of flows with $\hat{x}_n^i \neq 0$ at TS n, i.e., $\hat{\mathcal{I}}(n) = \{i : \hat{x}_n^i > 0, i \in \mathcal{I}\}$.

According to Algorithm 6, $x^1_{s_1} = \min(x_{s_1}, e_1)$. Thus $\hat{x}^1_{s_1} \leq x^1_{s_1}$. Then, we must have

$$\sum_{i \in \hat{\mathcal{I}}(s_1)\backslash\{1\}} \hat{x}^i_{s_1} + \sum_{t=s_1}^{d_1} \hat{x}^1_t \geq x^1_{s_1}. \tag{3.20}$$

This is because if $\sum_{i \in \hat{\mathcal{I}}(s_1)\backslash\{1\}} \hat{x}^i_{s_1} + \sum_{t=s_1}^{d_1} \hat{x}^1_t < x^1_{s_1}$, we could always increase $\hat{x}^1_t$ while keeping the rest allocation fixed without violating the constraints in (3.19).

Therefore, we can always transform the solution as follows:

1) Let $\tilde{x}^1_{s_1} = x^1_{s_1}$. Denote $\delta := \tilde{x}^1_{s_1} - \hat{x}^1_{s_1} > 0$, i.e., the extra energy allocated to demand f_1 at TS s_1 after the reallocation.

2) Sequentially reduce $\{\hat{x}^1_n\}_{n=s_1+1}^{d_1}$ greedily until the total reduced amount equals $\min\{\delta, \sum_{n=s_1+1}^{d_1} \hat{x}^1_n\}$. Denote the energy amounts after relocation as $\{\tilde{x}^1_n\}_{n=s_1+1}^{d_1}$, and $\delta_n := \hat{x}^1_n - \tilde{x}^1_n$. Then,

$$\sum_{t=s_1}^{d_1} \tilde{x}^1_t = \max\left\{x^1_{s_1}, \sum_{t=s_1}^{d_1} \hat{x}^1_t\right\} \leq e_1. \tag{3.21}$$

3) Decrease $\{\hat{x}^i_{s_1}\}_{i \in \hat{\mathcal{I}}(s_1)\backslash\{1\}}$ to $\{\tilde{x}^i_{s_1}\}_{i \in \hat{\mathcal{I}}(s_1)\backslash\{1\}}$ until the total reduced amount equals δ or $\tilde{x}^i_{s_1} = 0$ for all $i \in \hat{\mathcal{F}}(s_1)\backslash\{1\}$. Denoted $\epsilon^i_{s_1} := \hat{x}^i_{s_1} - \tilde{x}^i_{s_1}$.

4) Let $\rho = \delta - \min\{\delta, \sum_{i \in \hat{\mathcal{I}}(s_1)\backslash\{1\}} \hat{x}^i_{s_1}\}$, i.e., the extra energy allocated in s_1 after the reallocation. Note that $\rho \leq \sum_{n=s_1+1}^{d_1} \hat{x}^1_n$ due to (3.20). Then, for $i \in \hat{\mathcal{I}}(s_1)\backslash\{1\}$,

we increase $\hat{x}_n^i$ to $\tilde{x}_n^i$ by amount ϵ_n^i while satisfying the following conditions:

$$\sum_{n=s_1+1}^{d_1} \epsilon_n^i \leq \epsilon_{s_1}^i, \qquad \sum_{i \in \hat{\mathcal{I}}(s_1)\backslash\{1\}} \epsilon_n^i \leq \delta_n, \tag{3.22}$$

$$\sum_{n=s_1+1}^{d_1} \sum_{i \in \hat{\mathcal{I}}(s_1)\backslash\{1\}} \epsilon_{s_1}^i = \sum_{n=s_1+1}^{d_1} \delta_n - \rho. \tag{3.23}$$

The conditions in (3.22) ensure that the constraints in (3.19) are satisfied for any demand $i \in \hat{\mathcal{I}}(s_1)\backslash\{1\}$ and any TS $n = s_1 + 1, \ldots, d_1$, respectively. Besides, the condition in (3.23) ensures that $\sum_{n=s_1+1}^{d_1} \sum_{i \in \hat{\mathcal{I}}(s_1)} \hat{x}_n^i = \sum_{n=s_1+1}^{d_1} \sum_{i \in \hat{\mathcal{I}}(s_1)} \tilde{x}_n^i$.

If $x_{s_1}^1 < e_1$, we move to TS $s_1 + 1$, where $x_{s_1+1}^1 = \min(x_{s_1+1}, e_1 - x_{s_1}^1)$. Following similar arguments, we can show that the optimal solution can be adjusted to ensure $\tilde{x}_{s_1+1}^1 = x_{s_1+1}^1$. Once we have demand f_1 fulfilled, we move on to demand f_2. We can repeat this procedure until we have $\tilde{x}_n^i = x_n^i$ for all n, i. ■

Corollary 5. *Given $x_{I_1}, x_{I_2}, \cdots, x_{I_k}$, Algorithm 3 maximizes $x_{I_{k+1}}$ without violating (3.5), (3.7), and (3.8).*

Proof: Lemma 4 indicates that for given $x_{I_1}, x_{I_2}, \cdots, x_{I_k}$, Algorithm 4 and Algorithm 5 find the lower and upper bounds between $A(t)$ and $D(t)$ at TS I_{k+1} respectively without violating (3.5), (3.7), or (3.8). Thus, $x_{I_{k+1}}$ is maximized under Algorithm 3. ■

Theorem 1. *Given an admitted feasible set of demands $\mathcal{F}_{\mathcal{A}}$, the output of Algorithm 3 maximizes the profit of the charging station.*

The proof of Theorem 1 is provided in Appendix A.

Corollary 6. *An admitted demand set $\mathcal{F}_{\mathcal{A}}$ can be fulfilled if and only if Algorithm 3 outputs a schedule satisfying $\sum_{n=1}^{N} x_t = \sum_{i=1}^{m} e_i$.*

Proof: First, we note that Theorem 1 implies that if $\mathcal{F}_{\mathcal{A}}$ can be fulfilled, the output of Algorithm 3 must be feasible, thus the sum constraint is satisfied.

For the sufficiency, we point out that in each iteration of Algorithm 3, the constraints eqs. (3.5), (3.7) and (3.8) are always satisfied. Thus, if $\sum_{n=1}^{N} x_n = \sum_{i=1}^{k} e_i$, all constraints in Lemma 1 are satisfied. Therefore, the admitted demand set $\mathcal{F}_{\mathcal{A}}$ is feasible. ■

Remark: For the optimal scheduling, the traditional linear programming method, such as the ellipsoid method has complexity $\mathcal{O}(N^4m^4)$ [74]. In contrast, Algorithm 3 has complexity $\mathcal{O}(N^2)$ for the FIFO case, as we can recursively find the bounds by (3.17) and (3.18). For the general underloaded case, the complexity is $\mathcal{O}(N^2m)$, as we use DBA to find the bounds in each TS.

3.4 Admission Control and Optimal Scheduling

In general, all demands in $\mathcal{F}$ may not be fulfilled simultaneously. Thus, it is necessary to design an admission control mechanism to admit a subset of the demands, such that the admitted demands in $\mathcal{F}_{\mathcal{A}}$ can be fulfilled. However, as we will show it is NP-complete to find the optimal admitted demand set to maximize

the profit of the charging station.

Theorem 2. *For a given $\mathcal{F}$, identifying the optimal $\mathcal{F}_\mathcal{A}$ to maximize the profit in (4.1) and ensuring that all demands in $\mathcal{F}_\mathcal{A}$ are satisfied is NP-complete in general.*

The proof is provided in Appendix A.

Since the optimal admission control is NP-complete, we propose a heuristic algorithm to greedily decline the demands that are hard to be fulfilled under the total charging constraint. Let $r_i = \frac{e_i}{\sum_{n=s_i}^{d_i} p_{max}\tau/|\mathcal{F}_\mathcal{A}(n)|}$. Intuitively, $p_{max}\tau/|\mathcal{F}_\mathcal{A}(n)|$ is the amount of energy received by demand f_i assuming the maximum charging power is equally split among the active demands, and $\sum_{n=s_i}^{d_i} p_{max}\tau/|\mathcal{F}_\mathcal{A}(n)|$ is the corresponding total energy served to demand f_i. Thus, if r_i is way above one, it indicates that demand f_i is hard to be fulfilled, and should be declined. The Greedy Decline Algorithm is presented in Algorithm 7. It first initializes $\mathcal{F}_\mathcal{A}$ as the full demand set $\mathcal{F}$, and then greedily declines the demand with the largest r_i until all of the remaining demands are fulfilled under Algorithm 3. Finally, the DBA algorithm is adopted to obtain the optimal scheduling $\{x_n^i\}_{i,n}$. The complexity of Algorithm 7 is $\mathcal{O}(N^2m^2)$, as it invokes Algorithm 3 at most m times.

3.5 Online Policy

We now consider an online setting where the demands arrive sequentially, and the information of charging demand f_i is unavailable until its arrival time s_i. Thus,

Algorithm 7 Greedy Decline (GD)

Initialize: $\mathcal{F}_{\mathcal{A}} := \mathcal{F}$.
Invoke Algorithm 3.
while $\sum_{n=1}^{N} x_n < \sum_{i:f_i \in \mathcal{F}_{\mathcal{A}}} e_i$ **do**
    for $f_i \in \mathcal{F}_{\mathcal{A}}$ **do**
        $r_i = e_i / (\sum_{n=s_i}^{d_i} p_{max} \tau / |\mathcal{F}_{\mathcal{A}}(n)|)$
    end for
    $m = \max_{i:f_i \in \mathcal{F}_{\mathcal{A}}} r_i$
    $\mathcal{F}_{\mathcal{A}} = \mathcal{F}_{\mathcal{A}} \setminus f_m$
    Invoke Algorithm 3.
end while
$\{x_n^i\} \leftarrow \mathrm{DBA}(\mathcal{F}_{\mathcal{A}}, \{x_n\})$

once a new demand comes, the charging station will first decide whether to accept it or to decline it, and then scheduling the demand to maximize the revenue. We adopt similar idea as in the offline case to perform the admission control. For all set of demands arriving at the beginning of TS n, denoted as $\mathcal{F}_n$, we initialize the scheduling $x_n^i = 0, \bar{x}_n^i = 0$, for all $i \in \mathcal{F}_n, n \in \{s_i, s_i + 1, ..., d_i\}$. Then, $\mathcal{F}_n$ will be greedily declined into $\mathcal{F}_n'$ until they are feasible to be admitted simultaneously with the unfinished demands $\mathcal{F}_{\mathcal{A}}$. Thus, the admitted demand set will be updated as $\mathcal{F}_{\mathcal{A}} = \mathcal{F}_{\mathcal{A}} \cup \mathcal{F}_n'$. After the admission control, we update the schedule of $\mathcal{F}_{\mathcal{A}}$ at TS n. Finally, we update the remaining demanded energy from $\mathcal{F}_{\mathcal{A}}$ and goes to next TS $n + 1$. We note that if $\mathcal{F}_n = \emptyset$, we skip the admission control and rescheduling, and update the remaining demanded energy directly. This process is detailed in Algorithm 8.

Algorithm 8 Online GD

Initialize: $\mathcal{F}_{\mathcal{A}} = \emptyset$, $x_n^i = 0, \bar{x}_n^i = 0$.
for $n = 1, 2, ..., N$ **do**
  if $\mathcal{F}_n \neq \emptyset$ **then**
    $\mathcal{F}_{\mathcal{A}} = \mathcal{F}_{\mathcal{A}} \cup \mathcal{F}_n$
    Invoke Algorithm 3.
    while $\sum x_n < \sum e_i$ **do**
      for each $f_i \in \mathcal{F}_n$ **do**
        $r_i = e_i / (\sum_{n=s_i}^{d_i} p\max \cdot \tau / |\mathcal{F}_{\mathcal{A}}(n)|)$
      end for
      $m = \max_i r_i$
      $\mathcal{F}_{\mathcal{A}} = \mathcal{F}_{\mathcal{A}} \setminus f_m$
      Invoke Algorithm 3.
    end while
    $\{\bar{x}_n^i\} \leftarrow \text{DBA}(\mathcal{F}_{\mathcal{A}}, \{x_n\})$
    $x_n^k = \bar{x}_n^k, \forall k : f_k \subset \mathcal{F}_n$
  else
    $x_n^k = \bar{x}_n^k, \forall k : f_k \subset \mathcal{F}_n$
  end if
  for each demand $f_i \in \mathcal{F}_{\mathcal{A}}$ **do**
    $e_i = e_i - x_n^i$
  end for
end for
Return $\{x_n^i\}$;

3.6 Numerical Results

In this section, we evaluate the proposed algorithms through simulation. We compare our algorithms with two baseline algorithms, namely, the Earliest Deadline First (EDF) algorithm and the Least Laxity First (LLF) algorithm [75]. Under EDF, at each TS, demands with earlier deadlines have higher priority to be served. Under LLF, at each TS, the charge station assigns the demands with less laxity higher priority. We assume the time horizon of interest consists of 24 TSs, where

each TS lasts for 1 hour. The profit in each TS R_i is uniformly distributed in $[1, 8]$ dollar per kWh.

First, we study the profit with different total maximum charging power constraints, where the arrival TS s_i uniformly distributed in $[1, 23]$, and the duration between s_i and d_i are uniformly distributed in $[1, 24 - s_i]$. The demanded energy e_i is uniformly distributed in $[1, 6]$ kWh. With 50 randomly generated demands, we obtain the profit under the proposed algorithm and the baseline algorithms. Fig. 3.3 shows the average profit over 1000 samples when the charging power constraint varies from 1 to 15 kW. As shown in Fig. 3.3, our proposed algorithms achieve higher profit on average than EDF and LLF. As the total charging power constraint increases, the advantage of our algorithm becomes more significant, as more energy can be charged in TSs with higher profit rates. Table 3.1 shows the corresponding percentage of admitted demands that cannot be satisfied before their deadlines. As expected, EDF and LLF do the scheduling only based on deadline and laxity, respectively, thus cannot guarantee all demands be fulfilled, especially when the maximum charging power constraint is stringent. As expected, our algorithm guarantees that all admitted demands are fulfilled.

Next, we study the profit with different average demand durations. We fix the maximum charging power constraint as 9 kW, and evaluate the average profit when demand durations range from 1 to 10 hours. The results are shown in Fig. 3.4. As we observe, under our algorithm, when the average demand duration increases,

the average profit increases. This is because with longer demand duration, our algorithms have more flexibility to exploit the TSs with higher profit rates to increase the profit.

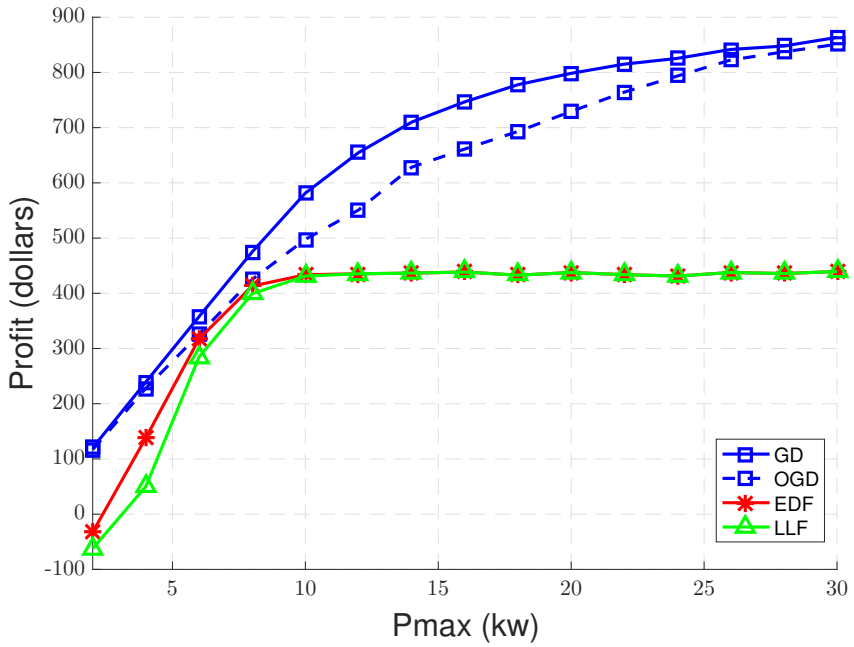


Figure 3.3: Average revenue under different charging constraint.

Table 3.1: Percentage of unsatisfied demands.

p_{max}	3	6	9	12	15
EDF	54.4	48.6	26.4	1.26	0.18
LLF	54.2	50.2	25.8	1.32	0.18

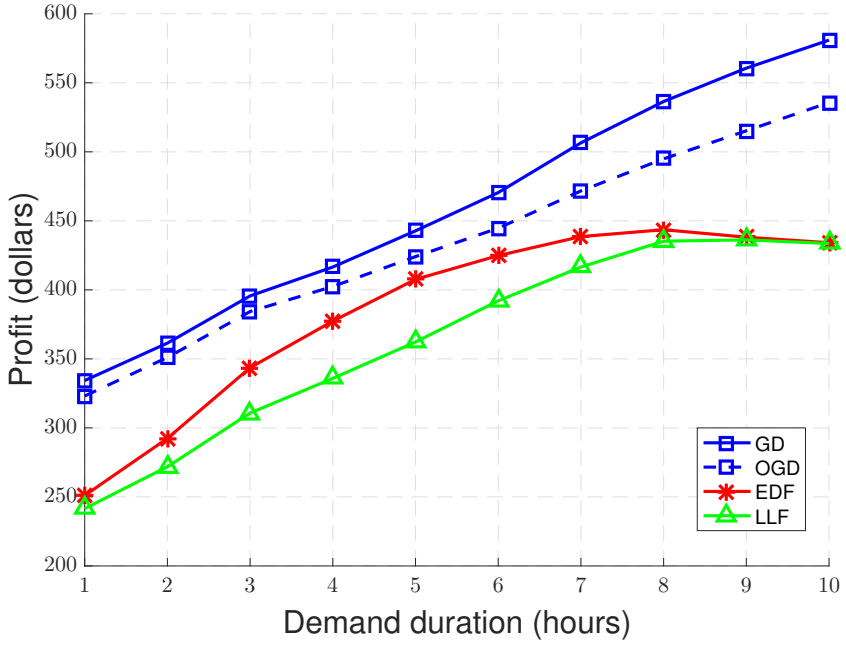


Figure 3.4: Average revenue under different charging constraint.

Then, we test our algorithms on real data. We use the electricity price of July 20, 2017 in NYC for our simulation. We set the length of a TS to be 15 minutes, and label the TS starting at $00:01$ am. The total number of TSs is 96 in a 24-hour setting. We assume the arrival TS of an EV is an i.i.d. random variable with the distribution specified in Table 3.2. The duration of each demand follows a uniform distribution over $[1, 40]$ TSs. The charging demands follow the setting in [69], which is a time-dependent exponential distribution. Considering a scenario that

Table 3.2: Distribution of EV arrival time.

k	[1,28]	[29,48]	[49,64]	[65,80]	[81,96]
$P[S_i = k]$	0.1/28	0.2/20	0.3/16	0.3/16	0.1/16

involves 200 vehicles per day, Fig. 3.5 shows the profit under different maximum charging power constraints. We can see that the proposed algorithms achieve higher profits than the baseline algorithms. Fig.3.6 shows the charing amount of different algorithms under real-time price. We can see that the proposed algorithms charged more energy when price is low and less when price is high.

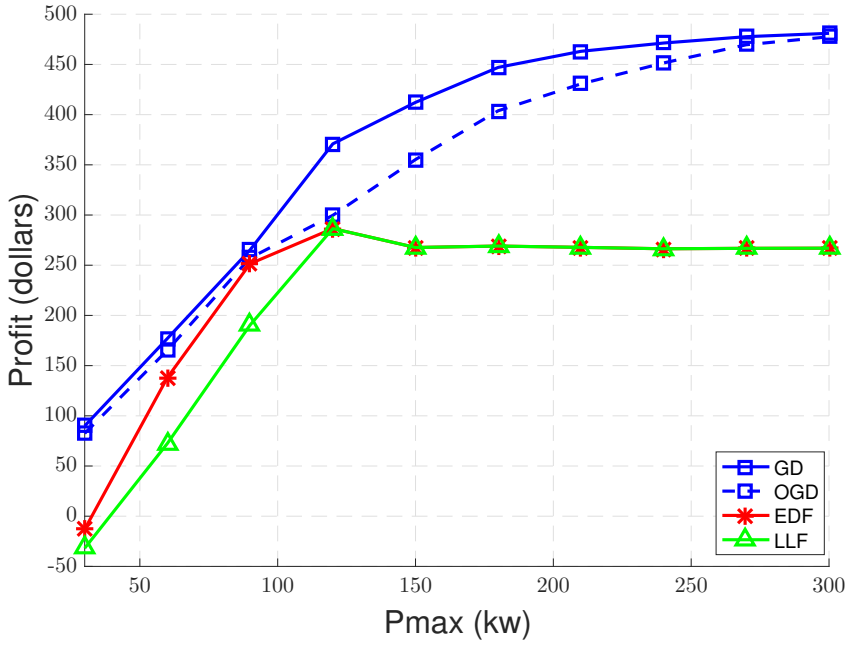


Figure 3.5: Profit versus maximum charging power constraints with real data.

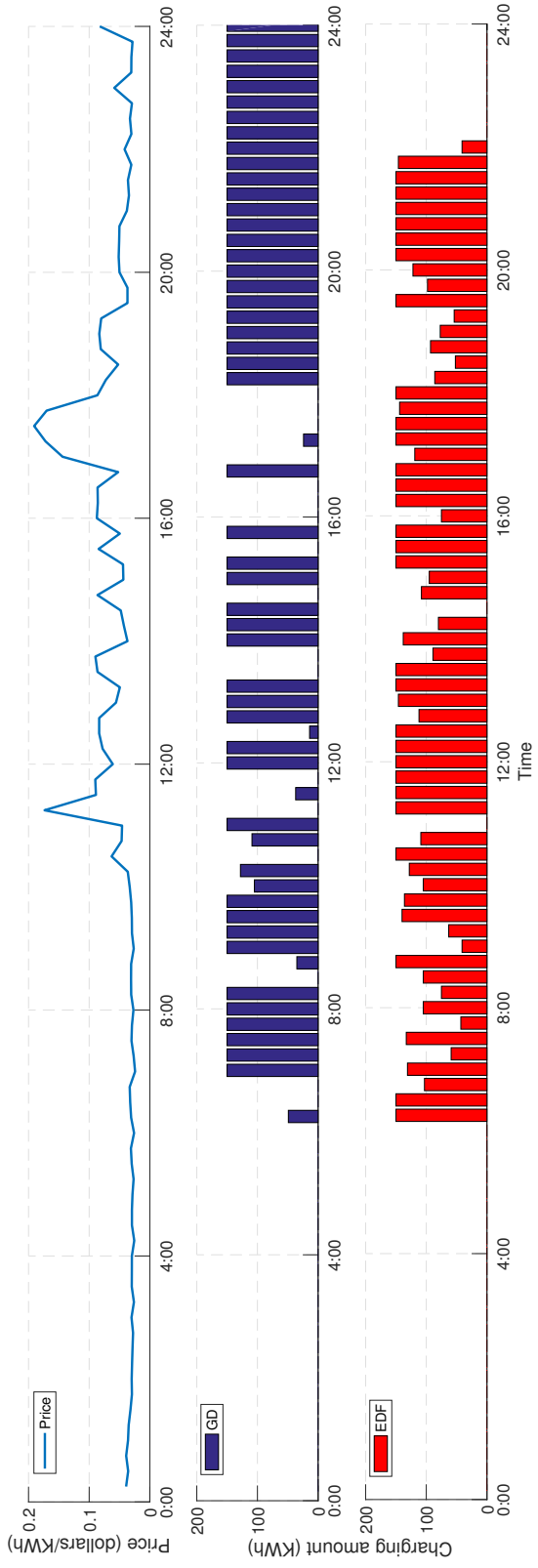


Figure 3.6: Charging amount under real time pricing.

3.7 Conclusions

We consider the admission control and scheduling of EVs at a charging station to maximize its revenue. We first propose a calculus based scheduling algorithm to maximize the revenue for a given admitted feasible demand set. We then show that the general admission control and scheduling problem is NP-complete, and develop a heuristic algorithm to first greedily do the admission control until the admitted demands are feasible, and then schedule the demands using the calculus based approach. We then consider an online setting and develop an online algorithm. Simulation results show that the proposed algorithms can achieve higher revenue than the baseline algorithms.

Chapter 4

Minimizing Age of Information under Link Capacity Constraint

In this chapter, we consider a scenario where a source continuously monitors an object and sends time-stamped status updates to a destination through a rate-limited link. We assume updates arrive randomly at the source according to a Bernoulli process. Due to the link capacity constraint, it takes multiple time slots for the source to complete the transmission of an update. Therefore, when a new update arrives at the source during the transmission of another update, the source needs to decide whether to skip the new arrival or to switch to it, in order to minimize the expected average age of information (AoI) at the destination. We start with the setting where all updates are of the same size, and prove that within a broadly defined class of online policies, the optimal policy should be a renewal policy, and has a sequential switching property. We then show that the optimal decision of the source in any time slot has threshold structures, and only depends

on the age of the update being transmitted and the AoI at the destination. We then consider the setting where updates are of different sizes, and show that the optimal Markovian policy also has a multiple-threshold structure. For each of the settings, we explicitly identify the thresholds by formulating the problem as a Markov Decision Process (MDP), and solve it through value iteration. Special structural properties of the corresponding optimal policy are utilized to reduce the computational complexity of the value iteration algorithm.

4.1 Introduction

Enabled by the proliferation of ubiquitous sensing devices and the pervasive wireless data connectivity, real-time status monitoring has become a reality in large-scale cyber-physical systems, such as power grids, manufacturing facilities, and smart transportation systems. However, the unprecedented high-dimensionality and generation rate of the sensing data also impose critical challenges on its timely delivery. In order to measure and ensure the freshness of the information available to the monitor, a metric called Age of Information (AoI) has been introduced and analyzed in various status updating systems [76]. Specifically, at time t, the AoI in the system is defined as $t - y(t)$, where $y(t)$ is the time stamp of the latest received update at the destination. Since AoI depends on data generation as well as queueing and transmission, it is fundamentally different from traditional network performance metrics, such as throughput and delay.

Modeling the status updating process as a queueing process, time-average AoI has been analyzed in systems with a single server [76–84], and multiple servers [85–89]. Peak Age of Information (PAoI) has been introduced and studied in [90–92]. AoI minimization has also been investigated, either by controlling the generation process of the updates [93–103], or by scheduling the transmission of updates that have already been generated [104–109]. Optimal status updating policy with knowledge of the server state has been studied in [93]. AoI-optimal sampling of a Wiener process is investigated in [94]. Under an energy harvesting setting, optimal status updating have been studied in [95–103]. Transmission scheduling for AoI minimization has been studied for broadcast channels [104–107], and for multiple access channels [108]. Age-optimal link scheduling in a multiple-source system with conflicting links is studied in [109], and the problem is shown to be NP-complete in general. Recently, a few works start to investigate the impact of service preemption on the time-average age in various status updating systems [77,110–116]. The common assumption is that new updates arrive at the source when it is in service (e.g., transmitting) of another update. Whether the source should preempt the current transmission thus would affect the age substantially. In [77], it shows that for an $M/M/1$ system with a buffer, the last-come-first-served (LCFS) with preemption discipline achieves lower time-average age than LCFS without preemption. In [110], it indicates that when the service time follows a Gamma distribution, last-generated-first-served (LGFS) with preemption may not

outperform LGFS without preemption. If the service times are i.i.d. and satisfy a New-Better-than-Used (NBU) distributional property, the non-preemptive LGFS policy is shown to be within a constant gap from the optimum age performance in [111]. Assuming exponential transmission time over network links, [112] shows that a preemptive LGFS policy results in smaller age than any other causal policy. Similar optimal properties of the preemptive and non-preemptive LGFS policies have been extended to a multi-flow setting in [113]. When the transmitter does not have a buffer, the problem becomes whether to drop a new update arrival or to drop the unfinished update and starts the new one. In [114], it studies an $M/G/1/1$ queue, and shows that when the updates are sent over an erasure channel, hybrid ARQ without preemption achieves smaller average AoI than the policy that always drops the unfinished updates. In [115], it considers an energy harvesting status updating system, and shows that if the service time is exponential and both the updating generation and energy harvesting process are Poisson, dropping unfinished updates achieves smaller age than dropping new arrivals when the system is in the "energy rich" regime. In [116], it focuses on stationary Markov and randomized policies that depend on the instantaneous AoI, and shows that whether to drop the new or old update depends on the service time distributions. In this paper, we investigate the age-optimal online transmission scheduling for a single link under the assumption that the link capacity is limited and each update takes multiple time slots to transmit. During the transmission of an update, new updates may

arrive. We assume the size of each update is available to the source once it arrives, thus it has accurate information about how many time slots it takes to deliver the update. Then, without a buffer, the source has to decide whether to switch to the new arrival, or to continue its current transmission and skip the new update, based on causally known update arrival profile (size and arrival time). We consider two possible scenarios, based on different assumptions on the sizes of the updates:

1) *Updates of uniform size.* In this case, the transmission time of each update is fixed and the AoI will be reset to the same value once the update is delivered successfully. We first prove that within a broadly defined class of online policies, the optimal policy should be a renewal policy, and the decision-making over each renewal interval only depends on the arrival time of the updates in that interval. Then, we show that the optimal renewal policy has a multiple-threshold structure, which enables us to formulate the problem as an MDP, and identify the thresholds numerically through structured value iteration.

2) *Updates of non-uniform sizes.* In this case, each update may take a different number of time slots to transmit. Thus the AoI will be reset to different values when an update is delivered successfully. Therefore, the optimal policy depends on the arrival times of the updates, as well as their sizes. To make the problem tractable, we restrict to stationary Markov policies where the decision of the transmitter depends on the AoI at the destination, the age of the update being transmitted, and the size of the new update. We show that the optimal policy exhibits certain

threshold structures along different dimensions of the system state. We then propose a structured value iteration algorithm to solve for the thresholds numerically.

4.2 System Model and Problem Formulation

We consider a single-link status monitoring system where the source keeps sending time-stamped status updates to a destination through a rate-limited link. We assume the time axis is discretized into time slots, labeled as $t = 1, 2, 3, \cdots$. At the beginning of time slot t, an update packet is generated and arrives at the source according to an independent and identically distributed (i.i.d.) Bernoulli process $\{a_t\}$ with parameter p. We consider the scenario where the size of each update is large compared with the link capacity, so that it takes multiple time slots to transmit. The distribution of the transmission time of the updates will be specified later. Similar to [104, 106, 107], we assume that at most one update can be transmitted during each time slot, and there is *no buffer* at the source to store any updates other than the one being transmitted. Therefore, once an update arrives at the source, it needs to decide whether to transmit it and drop the one being transmitted if there is any, or to drop the new arrival.

A status update policy is denoted as π, which consists of a sequence of transmission decisions $\{w_t\}$. We let $w_t \in \{0, 1\}$. Specifically, when $a_t = 1$, w_t can take both values 1 and 0: If $w_t = 1$, the source will start transmitting the new arrival in time slot t and drop the unfinished update if there is one. We term this as

switch; Otherwise, if $w_t = 0$, the source will drop the new arrival, and continue transmitting the unfinished update if there is one, or be idle otherwise. We term this as *skip*. When $a_t = 0$, we can show that dropping the update being transmitted is sub-optimal. Thus, we restrict to the policies under which w_t can only take value 0, i.e., to continue transmitting the unfinished update if there is one, or to idle.

Let S_n be the the time slot when an update is completely transmitted to the destination. Then, the inter-update delays can be denoted as $X_n := S_n - S_{n-1}$, for $n = 1, 2, \ldots$. Without loss of generality, we assume $S_0 = 0$. We note that since the update arrivals will either be dropped or transmitted immediately, the AoI after a completed transmission is reset to the transmission time of the delivered update, denoted as d_n. An example sample path of the AoI evolution under a given status update policy is shown in Fig. 4.1. As illustrated, some updates are skipped when they arrive, while others are transmitted partially or completely.

We use $N(T)$ to denote the total number of successfully delivered status updates over $(0, T]$. Define $R(T)$ as the cumulative AoI experienced by the system over $[0, T]$. Denote $R_n := (2d_n + X_n)X_n/2$, i.e., the total AoI experienced by the receiver over the nth epoch X_n. Then,

$$R(T) = \sum_{n=1}^{N(T)} R_n + \frac{1}{2}(d_n + T - S_{N(T)})(T - S_{N(T)}).$$

We focus on a set of *online* policies Π, in which the information available for determining w_t includes the decision history $\{w_i\}_{i=1}^{t-1}$, the update arrival profile

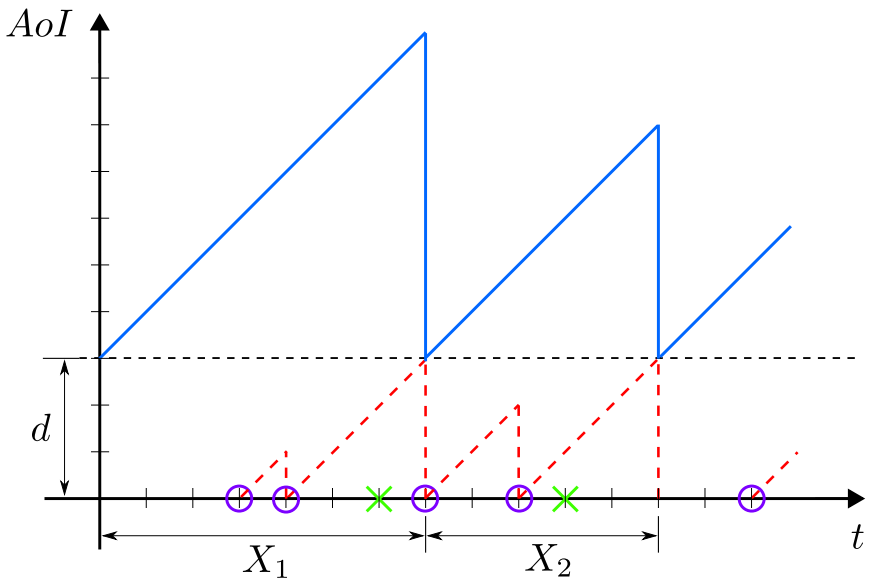


Figure 4.1: AoI evolution with $d = 3$. Circles represent transmitted updates, and crosses represent skipped ones. Red dashed curve indicates the transmitted portion of the corresponding update.

$\{a_i\}_{i=1}^{t}$, as well as the statistics of the update arrivals (i.e., arrival rate p and the distribution of update sizes). The optimization problem can be formulated as

$$\min_{\pi \in \Pi} \quad \limsup_{T \to \infty} \quad \mathbb{E}\left[\frac{R(T)}{T}\right], \tag{4.1}$$

where the expectation in the objective function is taken over all possible update arrival sample paths.

4.3 Updates of Uniform Size

In this section, we focus on the scenario where the updates are of the same size, and the required transmission time is equal to d time slots, where d is an integer greater than or equal to two.

4.3.1 Structure of the Optimal Policy

Consider the nth epoch, i.e., the duration between time slots $S_{n-1}+1$ and S_n under any online policy in Π. Let $a_{n,k}$ be the time slot when the kth update after S_{n-1} arrives, and let $x_{n,k} := a_{n,k} - S_{n-1}$. Denote the update arrival profile in epoch n as $\mathbf{x}_n := (x_{n,1}, x_{n,2}, \ldots)$. Then, we introduce the following definition.

Definition 1 (Uniformly Bounded Policy)**.** *Under an online policy $\pi \in \Pi$, if there exists a function $g(\mathbf{x})$, such that for any $\mathbf{x}_n = \mathbf{x}$, the length of the corresponding epoch X_n is upper bounded by $g(\mathbf{x})$, $\mathbb{E}[g^2(\mathbf{x}_n)] < \infty$, then this policy is a uniformly bounded policy.*

Denote the subset of uniformly bounded policies as Π'. Then, we have the following lemmas and theorems, which the proofs are provided in Appendix B.

Lemma 7. *Under any $\pi \in \Pi'$, it must have* $\lim_{T\to\infty} \frac{\mathbb{E}[X^2_{N(T)+1}]}{T}$.

Theorem 3. *The time-average AoI under any uniformly bounded policy $\pi \in \Pi'$ can always be improved by a renewal policy, under which $\{S_n\}_{n=1}^{\infty}$ form a renewal*

process, and the decision $\{w_t\}$ *over the nth renewal epoch only depends on* $\mathbf{x}_n$ *causally.*

Based on Theorem 3, in the following, we will focus on renewal policies that depend on $\mathbf{x}_n$ only.

Lemma 8. *If the source is idle when an update arrives, it should start transmitting the update immediately.*

Proof: We proof this lemma through contradiction. We assume the under the optimal policy π_0, there exists a sample path under which the source skip the file arrives at time slot i when the source is idle at the k^{th} epoch, i.e., $i < S_k$. Depending on the upcoming random arrivals, the sample path may evolve into different sample paths. Denote the set of such sample paths as $\mathcal{F}_i$, as they share the same history up to i. We then partition F_i into two subsets:

- $\mathcal{F}_{i,1}$: $i + d > S_k - d$.
- $\mathcal{F}_{i,2}$: $i + d \leq S_k - d$.

We now construct another policy π_1 under which the source will behave exactly the same as under π_0 for all sample paths not in $\mathcal{F}_i$. However, for the sample paths in $\mathcal{F}_{i,1}$, the source will first transmit the file arrives at i and then switch to the file with the arrival time $S_k - d$; For the sample paths in $\mathcal{F}_{i,2}$, the source will first transmit the file arrives at i, after the file has been received, the source will then transmit the file arrives at $S_k - d$. Therefore, under π_1, the sample paths in $\mathcal{F}_{i,1}$

will keep the same, while the sample paths in $\mathcal{F}_{i,2}$ will first drop to d at time slot $i+d$ and then will drop to d again at time slot $S_k - d$ instead of keep growing and only drop to d at time slot $S_k - d$, which will reduce the AoI.

Therefore, considering all possible sample paths under π_0 and π_1, we know π_1 achieves a lower expected average AoI than π_0, which contradicts with the assumption that π_0 is optimal. ■

We note the the optimal policy is to decide whether the transmitter should switch to a new update when it arrives.

Definition 2 (Sequential Switching Policy)**.** *A sequential switching (SS) policy is a renewal policy such that the source is allowed to switch to an update arriving at time slot t* ***only if*** *it has switched to all updates that arrive before t in the same epoch.*

Remark: The definition of SS policy implies that once a source skips a new update arrival at t, it will skip all of the upcoming update arrivals until it finishes the one being transmitted at t. We point out that an SS policy is in general different from threshold type of policies, as it does not impose any threshold structure on when the source should skip or switch to a new update arrival. We have the following observations.

Lemma 9. *The optimal renewal policy in Π' is an SS policy.*

Lemma 10. *Consider a renewal epoch under the optimal SS policy in Π'. If the source switches to an update at the ith time slot in that epoch, then, there exists a*

threshold τ_i, $\tau_i \geq i$, which depends on i only, such that if the next update arrives before or at the τ_ith time slot in that epoch, the source will switch to the new arrival; otherwise, it will skip any new updates until the end of the epoch.

The proofs of Lemma 9 and Lemma 10 are provided in Appendix B. There are proved through contradiction, i.e., if the optimal policy does not exhibit the SS and threshold structures, respectively, we can always construct another renewal policy to achieve smaller average AoI. The alternative policy is constructed in a sophisticated manner to ensure that the first moment of the length of each renewal interval stays the same as under the original policy while strictly reducing the second moment, thus achieving a smaller expected average AoI.

Based on Lemma 9 and Lemma 10, the structure of the optimal renewal policy is characterized in the following theorem.

Theorem 4. *Consider a renewal epoch under the optimal SS policy in Π'. There exists a sequence of thresholds $\tau_1 \geq \tau_2 \geq \cdots \geq \tau_K$, such that if the source switches to an update at the ith time slot in that epoch, $1 \leq i \leq K$, and the next update arrives before or at the τ_ith time slot in the epoch, the source will switch to the new arrival; Otherwise, if the next update arrives after the τ_ith time slot, or the update being transmitted arrives after the Kth slot in the epoch, the source will skip all upcoming arrivals until the end of the epoch.*

Theorem 4 indicates that the optimal decision of the source only depends on two parameters: the arrival time of the update being transmitted, and the arrival

time of the new update, both relative to the beginning of the renewal epoch.

4.3.2 MDP formulation

Motivated by the Markovian structure of the optimal policy in Theorem 4, we cast the problem as an MDP, and numerically search for the optimal thresholds $\tau_1, \tau_2, \cdots \tau_K$ and K as follows.

States: We define the state $\mathbf{s}_t := (\delta_t, u_t, a_t)$, where δ_t and u_t are the AoI in the system, and the age of the unfinished update, at the beginning of time slot t, respectively. Thus, for state $\mathbf{s}_t$, the correspond optimal threshold is $\tau_{\delta_t - d - u_t + 1}$. $a_t \in \{0, 1\}$ is the update arrival status. Then, $\delta_t \geq d$, $0 \leq u_t \leq d-1$, and the state space $\mathcal{S}$ can be determined accordingly.

Actions: $w_t \in \{0, 1\}$, as defined in Sec. 4.2.

Transition probabilities: Denote transition probability from state $\mathbf{s}_t$ to another state $\mathbf{s}_{t+1}$ under action w_t as $\pi(\mathbf{s}_{t+1}|\mathbf{s}_t, w_t)$. Then, if $w_t = 0$, i.e., the transmitter either continues its transmission, or stays idle, we have

$$\delta_{t+1} = \begin{cases} \delta_t + 1, & \text{if } u_t < d-1 \\ d, & \text{if } u_t = d-1 \end{cases}, \tag{4.2}$$

$$u_{t+1} = \begin{cases} u_t + 1, & \text{if } 0 < u_t < d-1 \\ 0, & \text{otherwise} \end{cases}. \tag{4.3}$$

If $w_t = 1$, i.e., the transmitter switches to a new update arriving at the beginning

of time slot t, then $\delta_{t+1} = \delta_t + 1$, $u_{t+1} = 1$.

Cost: Let $C(\mathbf{s}_t, w_t)$ be the instantaneous age under state $\mathbf{s}_t$, i.e., $C(\mathbf{s}_t, w_t) = \delta_t$.

Denote the relative value function as $V(\mathbf{s})$. Then the optimal Markovian policy to minimize the long-term average AoI is the solution to the following Bellman's equation [117]

$$J + V(\mathbf{s}) = \min_{w \in \mathcal{W}(\mathbf{s})} C(\mathbf{s}; w) + \mathbb{E}[V(\mathbf{s}')|\mathbf{s}, w]. \tag{4.4}$$

where J is the optimal average age, $\mathbf{s}'$ is the next system state when w is taken under state $\mathbf{s}$, and $\mathcal{W}(\mathbf{s})$ is the set of allowable actions for given state $\mathbf{s}$. $\mathcal{W}(\mathbf{s}) = \{0, 1\}$ if there is a new update arrival, and $\mathcal{W}(\mathbf{s}) = \{0\}$ otherwise.

Let the reference state be $\mathbf{s}_0 := (d, 0, 0)$. Then, the optimal policy can be determined through relative value iteration as follows:

$$V_{n+1}(\mathbf{s}) = \min_{w \in \mathcal{W}(\mathbf{s})} C(\mathbf{s}, w) + \sum_{s'} \pi(\mathbf{s}'|\mathbf{s}, w) V_n(\mathbf{s}') - V_n(\mathbf{s}_0), \tag{4.5}$$

where $V_0(\mathbf{s}) = 0$, $\forall \mathbf{s}$, and $V_n(\mathbf{s})$ converges to $V(\mathbf{s})$ as $n \to \infty$ [117].

To make the problem numerically tractable, we truncate the state space of the original MDP as $\mathcal{S}_m := \{\mathbf{s} \in \mathcal{S} : \delta \leq \delta_{max}\}$ by capping the AoI δ at δ_{max}, i.e., $[\delta]^+ = \min(\delta, \delta_{max})$. It can be shown that when δ_{max} is sufficiently large, the optimal policy for the truncated MDP is identical to that of the original MDP [118].

In order to further reduce the computational complexity, we leverage the multi-

Algorithm 9 Structured Value Iteration: Uniform Update Size

Initialize: $V_0(\mathbf{s}) = 0, \forall \mathbf{s} \in \mathcal{S}_m$.
for $i = 0 : n$ **do**
  for $\forall \mathbf{s} \in \mathcal{S}_m$ **do**
    if $a = 0$ **then**
      $w^*(\mathbf{s}) = 0$;
    else if $\exists \delta' < \delta, w^*(\delta', u, 1) = 0$ **then**
      $w^*(\mathbf{s}) = 0$;
    else if $\exists u' < u, w^*(\delta, u', 1) = 1$ **then**
      $w^*(\mathbf{s}) = 1$;
    else
      $w^*(\mathbf{s}) = \arg\min_{w \in \{0,1\}} C(\mathbf{s}, w) + \sum_{\mathbf{s}'} \pi(\mathbf{s}'|\mathbf{s}, w) V_i(\mathbf{s}')$
    end if
    $V_{i+1}(\mathbf{s}) = C(\mathbf{s}; w^*(\mathbf{s})) + \sum_{\mathbf{s}'} \pi(\mathbf{s}'|\mathbf{s}, w) V_i(s') - V_i(\mathbf{s}_0)$
  end for
end for
return $\{w^*(\mathbf{s})\}, \{V(\mathbf{s})\}$.

threshold structure of the optimal policy during the value iteration procedure, as detailed in the structured value iteration algorithm in Algorithm 9. With the multiple-threshold structure, Algorithm 9 does not need to seek the optimal action by equation (4.5) for all states in each iteration as the traditional value iteration algorithm does. Specifically, if the optimal action for a state $(\delta', u, 1)$ is to *skip* the new arrival, the optimal action for state $(\delta, u, 1)$, $\delta > \delta'$, must be to skip as well. Similarly, if the optimal action for a state $(\delta, u', 1)$ is to switch to the new arrival, the optimal action for state $(\delta, u, 1)$, $u < u'$, must be to switch. Thus, the computational complexity can be significantly reduced. Besides, since the truncated MDP is a unichain, Algorithm 9 converges in a finite number of iterations [119].

4.3.3 A special case

In thus subsection, we consider the special case that $d = 2$, and obtain the thresholds explicitly. We note for this case, the maximum age of the unfinished update is 1. Thus the state space is $\mathcal{S} = \{(2, 0, \cdot), (3, 0, \cdot), (3, 1, \cdot), (4, 0, \cdot), (4, 1, \cdot) \cdots\}$. Therefore, according to Theorem 4, there exists a single threshold $\bar{\delta}$ such that if $\delta < \bar{\delta}$, for state $(\delta, 1, 1)$, the optimal policy is to switch to the new arrival; otherwise, the optimal policy is to keep transmitting the current update for another time slot and complete the transmission. Thus the post-action states (δ, l) forms a discrete time Markov chain (DTMC), as shown in Fig. 4.2.

Theorem 5. *For $d = 2$, the long term average AoI with a given $\bar{\delta}$ is*

$$f(\bar{\delta}) = \frac{2p - 1 + (p\bar{\delta} + 1)(1-p)^{\bar{\delta}}p^2 + p^{\bar{\delta}+2}(p\bar{\delta} - \bar{\delta} - 1)}{p(1-p)(2p - 1 - p^{\bar{\delta}+1} + p^2(1-p)^{\bar{\delta}-1})}. \tag{4.6}$$

Proof: Based on the DTMC, we have the probability transition:

$$P(\delta, l) = \begin{cases} (1-p)P(\delta - 1, 0), & \text{if } l = 0, \delta = 3, \cdots; \\ (1-p)\sum_{i=3}^{\bar{\delta}-1}(P(i, 1) + \sum_{i=\bar{\delta}}^{\infty} P(i, 1)), & \text{if } l = 0, \delta = 2; \\ p(P(\delta - 1, 1) + P(\delta - 1, 0)), & \text{if } l = 1, \delta = 4, \cdots, \bar{\delta}; \\ pP(\delta - 1, 0) & \text{if } l = 1, \delta = 3, \bar{\delta} + 1, \bar{\delta} + 2 \cdots. \end{cases} \tag{4.7}$$

Thus the steady distribution of the DTMC is

$$\pi(\delta,l)=\begin{cases}\frac{p(1-p)^{\delta-1}(2p-1)}{2p-1-p^{\bar{\delta}+1}+p^2(1-p)^{\bar{\delta}-1}}, & \text{if } l=0,\delta=2,3,\cdots;\\ \frac{p^2(1-p)(p^{\delta-2}-(1-p)^{\delta-2})}{2p-1-p^{\bar{\delta}+1}+p^2(1-p)^{\bar{\delta}-1}}, & \text{if } l=1,\delta=3,\cdots,\bar{\delta};\\ \frac{p^2(1-p)^{\delta-2}(2p-1)}{2p-1-p^{\bar{\delta}+1}+p^2(1-p)^{\bar{\delta}-1}}. & \text{if } l=1,\delta=\bar{\delta}+1,\cdots.\end{cases} \tag{4.8}$$

Thus the average age of the DTMC is

$$\sum_{\delta=2}^{\infty}\delta\pi(\delta,l)=\frac{2p-1+(p\bar{\delta}+1)(1-p)^{\bar{\delta}}p^2+p^{\bar{\delta}+2}(p\bar{\delta}-\bar{\delta}-1)}{p(1-p)(2p-1-p^{\bar{\delta}+1}+p^2(1-p)^{\bar{\delta}-1})}. \tag{4.9}$$

■

Based on the Theorem 5, we can numerically find the optimal threshold to minimize the average AoI.

Note: Since $d=2$, the minimum AoI in the system equals 2. Thus $\bar{\delta}\geq 3$. When $\bar{\delta}=3$, the average AoI is $\frac{2p^2+2p+1}{p+1}$, which correspond to the no-preemptive policy; When $\bar{\delta}$ goes to infinity, the average age is $\frac{1}{p(1-p)}$ which correspond to the preemptive policy.

Corollary 11. *For any fixed threshold $\bar{\delta}$, the average age goes to infinity when p goes to 0, and it goes to $\frac{2+\bar{\delta}}{2}$ when p goes to 1.*

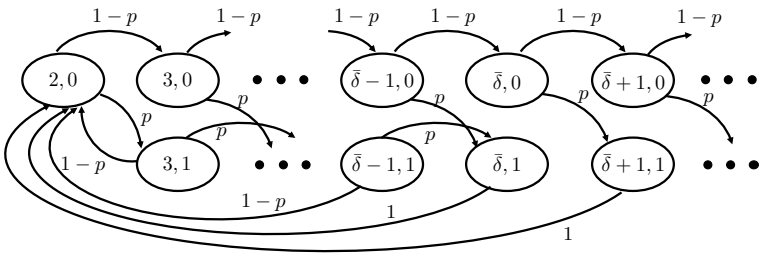


Figure 4.2: The post-action state under the threshold structure policy forms a DTMC.

Proof: The first part can be derived from (4.6) directly. When $p = 1$, the file arrivals at each TS. Under the threshold structure policy with threshold $\bar{\delta}$, the DTMC becomes a single loop, which is shown is Fig.4.3. The steady-distribution of this DTMC is

$$\pi(\delta, l) = \begin{cases} \frac{1}{\bar{\delta}-1} & \text{if } l = 0, \delta = 2;\ l = 1, \delta = 3, ..., \bar{\delta}; \\ 0 & \text{otherwise;} \end{cases} \tag{4.10}$$

Thus the average age is

$$\sum_{\delta=2}^{\infty} \delta\pi(\delta, l) = \frac{2 + \bar{\delta}}{2}, \tag{4.11}$$

the optimal threshold is $\bar{\delta} = 3$. ■

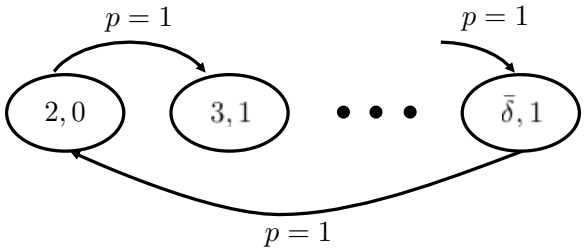


Figure 4.3: The post-action state when $p = 1$.

4.4 Updates of Non-uniform Sizes

In this section, we consider the scenario where the size of each update is non-uniform, and the required transmission time is a random variable following a known distribution. Specifically, we assume the required transmission time of each update is an i.i.d random variable following a probability mass function (PMF) f_b over a bounded support in $\mathbb{Z}_+$. For ease of exposition, we assume each update takes at least two time slots to transmit. Once a new update arrives at the source, its size is revealed to the transmitter immediately. Our objective is to decide whether to switch to the new update or to skip it, based on causal observations and the statistics of the update arrivals.

Compared with the uniform update size case, the source should also track the various sizes of the updates and take these into consideration when making decisions. In order to make the problem tractable, we restrict to Markovian polices where the decision at any time slot t depends on the AoI at the destination, denoted as δ_t, the remaining transmission time of the update being transmitted, denoted as l_t, the total required transmission time of the update being transmitted, denoted as c_t, and the total required transmission time of the new arrival, denoted as b_t. If no new update arrives at the time slot t, $b_t = 0$; Otherwise b_t follows distribution f_b.

Denote the system state at time slot t as $\mathbf{s}_t := (\delta_t, l_t, c_t, b_t)$. We note that $0 \leq l_t \leq c_t - 1$. Then, after the source takes an action $w_t \in \{0, 1\}$, the state at

time slot $t+1$ will change as follows. If $w_t = 0$, the source either continues its transmission and skips the new update, or stays idle. Thus,

$$\delta_{t+1} = \begin{cases} \delta_t + 1, & \text{if } l_t \neq 1, \\ c_t, & \text{if } l_t = 1, \end{cases} \tag{4.12}$$

$$l_{t+1} = \begin{cases} l_t - 1, & \text{if } l_t > 1, \\ 0, & \text{if } l_t = 0, 1, \end{cases} \tag{4.13}$$

$$c_{t+1} = \begin{cases} c_t, & \text{if } l_t > 1, \\ 0, & \text{if } l_t = 0, 1. \end{cases} \tag{4.14}$$

If $b_t \neq 0$ and $w_t = 1$, i.e., the source switches to the new arrival, $\delta_{t+1} = \delta_t + 1$, $l_{t+1} = b_t - 1$, $c_{t+1} = b_t$.

Let $C(\mathbf{s}_t; w_t)$ be the immediate cost under state $\mathbf{s}_t$ with action w_t. Similar to the uniform size case, we let $C(\mathbf{s}_t; w_t) = \delta_t$.

4.4.1 Structural Properties of Optimal Policy

In order to obtain some structural properties of the optimal policy, in this section, we will first introduce an infinite horizon α-discounted MDP as follows:

$$V^{\alpha}(\mathbf{s}) = \min_{w \in \mathcal{W}(\mathbf{s})} C(\mathbf{s}; w) + \alpha \mathbb{E}[V^{\alpha}(\mathbf{s}')|\mathbf{s}, w], \tag{4.15}$$

where $0 < \alpha < 1$. It has been shown that the optimal policy to minimize the average AoI can be obtained by solving (4.15) when $\alpha \to 1$ [117]. In order to identify the structural properties of the optimal policy, we start with the following value iteration formulation:

$$V_{n+1}^{\alpha}(\mathbf{s}) = \min_{w \in \mathcal{W}(\mathbf{s})} C(\mathbf{s}; w) + \alpha \mathbb{E}[V_n^{\alpha}(\mathbf{s}')|\mathbf{s}, w], \tag{4.16}$$

where $V_0^{\alpha}(\mathbf{s}) = 0, \forall \mathbf{s}$.

Assume $\mathbf{s} := (\delta, l, c, b)$. If $b \neq 0$, denote

$$Q_n(\mathbf{s}; w) := C(\mathbf{s}; w) + \alpha \mathbb{E}[V_n^{\alpha}(\mathbf{s}')|\mathbf{s}, w], \tag{4.17}$$

$$\Delta Q_n^{w;w'}(\mathbf{s}) := Q_n(\mathbf{s}; w) - Q_n(\mathbf{s}; w'). \tag{4.18}$$

Then, $V_{k+1}^{\alpha}(\mathbf{s}) = \min_{w \in \{0,1\}} Q_k(\mathbf{s}; w)$. We have

$$Q_k(\mathbf{s}; 0) = \begin{cases} \delta + \alpha \mathbb{E}[V_k^{\alpha}(\delta + 1, (l-1)^+, c, \bar{b})], & \text{if } l \neq 1, \\ \delta + \alpha \mathbb{E}[V_k^{\alpha}(c, 0, 0, \bar{b})], & \text{if } l = 1, \end{cases} \tag{4.19}$$

and

$$Q_k(\mathbf{s}; 1) = \delta + \alpha \mathbb{E}[V_k^{\alpha}(\delta + 1, b - 1, b, \bar{b})], \tag{4.20}$$

where the expectation is taken with respect to $\bar{b}$, and $(x)^+ = \max\{x, 0\}$. We adopt

$(l-1)^+$ in (4.19) to indicate that if $l = c = 0$, and $w = 0$, the system will remain idle.

If $b = 0$, $V_{k+1}^{\alpha}(\mathbf{s}) = Q_k(\mathbf{s}; 0)$, where $Q_k(\mathbf{s}; 0)$ follows the same form of (4.19).

We have the following observations:

Lemma 12. *Let* $\mathbf{s} := (\delta, l, c, b)$. *Then,* $V_n^{\alpha}(\mathbf{s})$ *is monotonically increasing in* δ, *in* c *for* $c > 0$, *and in* b *for* $b > 0$, $\forall n$.

Proof: We first prove the monotonicity in δ through induction. First, it holds when $n = 0$. Assume it holds for $n = k$. Then, it suffices to show that it holds for $n = k + 1$ as well. If $b > 0$, $V_{k+1}^{\alpha}(\mathbf{s}) = \min_{w \in \{0,1\}} Q_k(\mathbf{s}; w)$. We note that $Q_k(\mathbf{s}; 1)$ is an increasing function in δ according to the induction assumption. If $l = 1$, $Q_k(\mathbf{s}; 0)$ is increasing in δ based on its definition in (4.19). If $l \neq 1$, $Q_k(\mathbf{s}; 0)$ is increasing in δ according to the induction assumption. Thus, $V_{k+1}^{\alpha}(\mathbf{s})$ must monotonically increase in δ, as taking the minimum of $Q_k(\mathbf{s}; 0)$ and $Q_k(\mathbf{s}; 1)$ preserves the monotonicity. If $b = 0$, $V_{k+1}^{\alpha}(\mathbf{s}) = Q_k(\mathbf{s}; 0)$, which is monotonically increasing in δ as well.

Next, we prove the monotonicity in c for $c > 0$. We prove it through induction as well. It holds when $n = 0$. Assume it is true for $n = k$. We note that $Q_k(\mathbf{s}; 1)$ is independent of c when $b > 0$. Besides, if $l = 1$, $Q_k(\mathbf{s}; 0)$ is increasing in c based on the monotonicity of $V_n^{\alpha}(\mathbf{s})$ in δ. If $1 < l < c$, $Q_k(\mathbf{s}; 0)$ is increasing in c based on to the induction assumption. Thus $V_{k+1}^{\alpha}(\mathbf{s})$ is increasing in c for $c > 0$.

In order to show the monotonicity of $V_n^{\alpha}(\mathbf{s})$ in b for $b > 0$, we first show that $Q_n(\mathbf{s}; 1)$ is increasing in b for $b > 0$, $\forall n$. Based on the definition of $Q_n(\mathbf{s}; 1)$ in

(4.20), it suffices to show $V_n^\alpha(\delta, b-q, b, \bar{b})$ is increasing in b, for $1 \leq q \leq b$. We note it holds when $n = 0$. Assume $V_k^\alpha(\delta, b-q, b, \bar{b})$ is increasing in b. Then, we will show that $V_{k+1}^\alpha(\delta, b-q, b, \bar{b})$ is increasing in b as well. We note that if $\bar{b} \neq 0$, $Q_k((\delta, b-q, b, \bar{b}); 1)$ is independent with b. Besides, when $q = b-1$, $Q_k((\delta, b-q, b, \bar{b}); 0) = \delta + \alpha\mathbb{E}[V_k^\alpha(b, 0, 0, \bar{b})]$, which is increasing in b according to the monotonicity of $V_k^\alpha(b, 0, 0, \bar{b})$; when $1 \leq q < b-1$, $Q_k((\delta, b-q, b, \bar{b}); 0) = \delta + \alpha\mathbb{E}[V_k^\alpha(\delta+1, b-q-1, b, \bar{b})]$, which is increasing in b based on the induction assumption. Thus, $V_{k+1}^\alpha(\delta, b-q, b, \bar{b})$ is increasing in b after taking the minimum of $Q_k((\delta, b-q, b, \bar{b}); 0)$ and $Q_k((\delta, b-q, b, \bar{b}); 1)$. The monotonicity of $Q_n(\mathbf{s}; 1)$ in b for $b > 0$ is thus established.

Since $Q_n(\mathbf{s}; 0)$ is independent of b, while $Q_n(\mathbf{s}; 1)$ is increasing in b when $b > 0$, after taking the minimum of them, $V_n^\alpha(\mathbf{s})$ is increasing in b for $b > 0$ as well. ■

Lemma 13. $\mathbb{E}[V_n^\alpha(\delta, l_1, c_1, \bar{b})] \leq \mathbb{E}[V_n^\alpha(\delta, l_2, c_2, \bar{b})]$ *if and only if* $Q_{n-1}((\delta, l_1, c_1, .); 0) \leq Q_{n-1}((\delta, l_2, c_2, .); 0)$, $\forall n \geq 1$.

Proof: First, we note that

$$\begin{aligned}
&\mathbb{E}[V_n^\alpha(\delta, l, c, \bar{b})] \\
&= (1-p)Q_{n-1}((\delta, l, c, 0); 0) \\
&\quad + p\mathbb{E}\left[\min_{w \in \{0,1\}} Q_{n-1}((\delta, l, c, \bar{b}); w) \middle| \bar{b} \neq 0\right]. \qquad (4.21)
\end{aligned}$$

Besides, for $\bar{b} \neq 0$,

$$Q_{n-1}((\delta, l_1, c_1, \bar{b}); 1) = Q_{n-1}((\delta, l_2, c_2, \bar{b}); 1). \tag{4.22}$$

If

$$Q_{n-1}((\delta, l_1, c_1, .); 0) \leq Q_{n-1}((\delta, l_2, c_2, .); 0), \tag{4.23}$$

then, according to (4.21),

$$\mathbb{E}[V_n^{\alpha}(\delta, l_1, c_1, \bar{b})] \leq \mathbb{E}[V_n^{\alpha}(\delta, l_2, c_2, \bar{b})]. \tag{4.24}$$

The sufficiency is thus established.

On the other hand, if

$$Q_{n-1}((\delta, l_1, c_1, .); 0) > Q_{n-1}((\delta, l_2, c_2, .); 0), \tag{4.25}$$

then for every possible $\bar{b} \neq 0$, we must have

$$\min_{w \in \{0,1\}} Q_{n-1}((\delta, l_1, c_1, \bar{b}); w) \geq \min_{w \in \{0,1\}} Q_{n-1}((\delta, l_2, c_2, \bar{b}); w).$$

Based on (4.21) and the assumption that $0 < p < 1$, we have

$$\mathbb{E}[V_n^\alpha(\delta, l_1, c_1, \bar{b})] > \mathbb{E}[V_n^\alpha(\delta, l_2, c_2, \bar{b})], \tag{4.26}$$

which proves the necessity of the condition. ■

Corollary 14. *For states with* $l_1, l_2 > 1$, $\mathbb{E}[V_n^\alpha(\delta, l_1, c_1, \bar{b})] \leq \mathbb{E}[V_n^\alpha(\delta, l_2, c_2, \bar{b})]$ *if and only if* $\mathbb{E}[V_{n-1}^\alpha(\delta+1, l_1-1, c_1, \bar{b})] \leq \mathbb{E}[V_{n-1}^\alpha(\delta+1, l_2-1, c_2, \bar{b})]$, $\forall n \geq 1$.

Proof: Since $Q_{n-1}((\delta, l, c, .); 0) = \delta + \alpha\mathbb{E}[V_{n-1}^\alpha(\delta+1, l-1, c, \bar{b})]$ when $l > 1$, $Q_{n-1}((\delta, l_1, c_1, .); 0) \leq Q_{n-1}((\delta, l_2, c_2, .); 0)$ is equivalent to

$$\mathbb{E}[V_{n-1}^\alpha(\delta+1, l_1-1, c_1, \bar{b})] \leq \mathbb{E}[V_{n-1}^\alpha(\delta+1, l_2-1, c_2, \bar{b})].$$

The corollary is thus proved based on Lemma 13. ■

Corollary 15. *For states with* $l_1 > 1$, $\mathbb{E}[V_n^\alpha(\delta, l_1, c_1, \bar{b})] \leq \mathbb{E}[V_n^\alpha(\delta, 0, 0, \bar{b})]$ *if and only if* $\mathbb{E}[V_{n-1}^\alpha(\delta+1, l_1-1, c_1, \bar{b})] \leq \mathbb{E}[V_{n-1}^\alpha(\delta+1, 0, 0, \bar{b})]$, $\forall n \geq 1$.

Corollary 15 can be proved in a way similar to the proof of Corollary 14, and is thus omitted.

Based on Lemmas 12, 13 and Corollaries 14, 15, we have the following theorem.

Theorem 6. *Denote* $\mathbf{s} = (\delta, l, c, b)$ *and assume* $b > 0$. *The optimal policy for the* α*-discounted MDP has the following structure:*

(a) If the optimal action for $\mathbf{s}$ *is to switch, then the optimal action for any state* (δ, l, c, b'), $0 < b' < b$, *is to switch as well.*

(b) If $0 < b \leq l$, *then the optimal action for* $\mathbf{s}$ *is to switch to the new update.*

(c) If the optimal action for $\mathbf{s}$ *is to switch, then for any* $c' > c$, *the optimal action for state* $\mathbf{s}' \triangleq (\delta, l, c', b)$ *is to switch as well.*

(d) If $l = c = 0$, *then the optimal action for for* $\mathbf{s}$ *is to switch to the new update.*

(e) If the optimal action for $\mathbf{s}$ *is to skip, then the optimal action for state* $(\delta + 1, l, c, b)$ *is to skip as well.*

The proof of Theorem 6 is provided in Appendix B. Theorem 6(a) indicates the threshold structure on the size of the new update arrival, i.e., the source prefers to switch to a new update if its size is small, and will skip it if its size is large. Theorem 6(b) is a consequence of Theorem 6(a). Theorem 6(c) shows that there exists a threshold on the size of the update being transmitted, i.e., the source prefers to drop updates with larger sizes and switch to new updates. Theorem 6(d) says that the source should immediately start transmitting the new update arrival if it has been idle, which is consistent with Lemma 8 for the uniform update size case. Theorem 6(e) essentially indicates the threshold structure on the instantaneous AoI at the destination: that the source prefers to skip new updates when the AoI is large, as it is in more urgent need to complete the current transmission and reset the AoI to a smaller value.

We point out that all of the structural properties of the optimal policy derived for the α-discounted problem hold when $\alpha \to 1$ [106]. Thus, the optimal policy for the time-average problem also exhibits similar structures.

4.4.2 Structured Value Iteration

Following an approach similar to the uniform update size case, we leverage the structural properties of the optimal policy and develop a structured value iteration algorithm to obtain the thresholds numerically. The detailed algorithm is presented in Algorithm 9.

Algorithm 10 Structured Value Iteration: Non-uniform Update Sizes

Initialize: $V_0(\mathbf{s}) = 0, \forall \mathbf{s} \in \mathcal{S}_m$.
for $i = 0 : n$ **do**
  for $\forall \mathbf{s} \in \mathcal{S}_m$ **do**
    if $d = 0$ **then**
      $w^*(\mathbf{s}) = 0$;
    else if $l, c = 0$ **then**
      $w^*(\mathbf{s}) = 1$;
    else if $\exists \delta' < \delta, w^*(\delta', l, c, b) = 0$ **then**
      $w^*(\mathbf{s}) = 0$;
    else if $\exists c' > c, w^*(\delta, l, c', b) = 0$ **then**
      $w^*(\mathbf{s}) = 0$;
    else if $\exists d' > b, w^*(\delta, l, c, b') = 1$ **then**
      $w^*(\mathbf{s}) = 1$;
    else
      $w^*(\mathbf{s}) = \arg\min_{w \in \{0,1\}} C(\mathbf{s}; w) + \sum_{\mathbf{s}'} \pi(\mathbf{s}'|\mathbf{s}, w) V_i(\mathbf{s}')$
    end if
    $V_{i+1}(\mathbf{s}) = C(\mathbf{s}, w^*(\mathbf{s})) + \sum_{\mathbf{s}'} \pi(\mathbf{s}'|\mathbf{s}, w) V_i(\mathbf{s}') - V_i(\mathbf{s}_0)$
  end for
end for
return $\{w^*(\mathbf{s})\}, \{V(\mathbf{s})\}$.

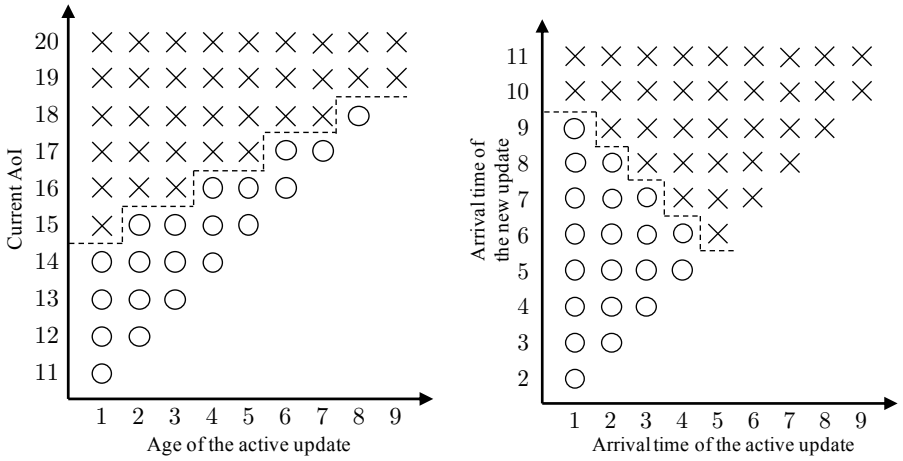
4.5 Numerical results

In this section, we numerically search for the optimal policies for both uniform and non-uniform update size cases according to Algorithm 9 and Algorithm 10, respectively. For both truncated MDPs, we set $\delta_{max} = 1000$ and the number of iterations to be $10,000$.

4.5.1 Updates of Uniform Size

First, we focus on the uniform update size case. We set $d = 10$, $p = 0.07$. Fig. 4.4(a) shows the optimal action for each state $(\delta, u, 1)$. We note the monotonicity of the thresholds in both δ and u. We then plot the optimal action for each pair of arrival time of the update being transmitted (i.e., active update) and that of the new arrival in a renewal epoch in Fig. 4.4(b). We note that the thresholds $\tau_1 = 9$, $\tau_2 = 8$, $\tau_3 = 7$, and $\tau_4 = 6$. They are monotonically decreasing, as predicted by Theorem 4. When the update being transmitted arrives later than the third time slot in that epoch, all upcoming updates will be skipped.

Then, we evaluate the time-average AoI under the optimal policy identified by Algorithm 9 and two baseline policies, namely, the Always Skip and Always Switch policies, over $10,000$ time slots. Under the Always Skip policy, the source will never switch to any new update arrival until it finishes the one being transmitted, while under the Always Switch policy, the source will always switch to new updates upon



(a) (b)

Figure 4.4: The optimal policy when $p = 0.07$, $d = 10$. Circles represent *switch*, while crosses represent *skip*.

their arrivals. As we observe in Fig. 4.5, the performance differences between those three policies are negligible when p is close to 0. This is because when p is small, the source won't receive a new update before it finishes transmitting the current update with high probability, thus the source behaves almost identically under all three policies. As p increases, the performances of the optimal policy and the Always Skip policy are still very close to each other, while the Always Switch policy renders the highest AoI. To distinguish the performances of the Always Skip policy and the optimal policy, we plot the their performance gap on time-average AoI in Fig. 4.6. As we note, as p increases, the performance gap first increases and then decreases to zero. This is because when p is not very large, the source still needs to preempt the transmission of an update occasionally in order to regulate the inter-update delays under the optimal policy; and when p is sufficiently large, this becomes unnecessary, as it will have a new update arrival soon after a successful

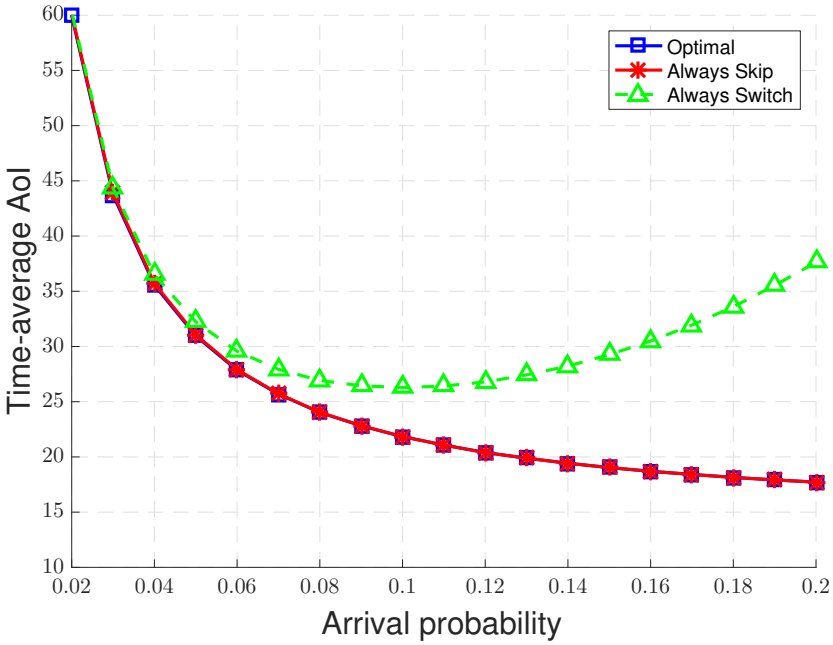


Figure 4.5: Average AoI with $d = 10$.

update. Thus, the optimal policy becomes the same as the Always Skip policy in this regime. It is interesting to note the surprisingly small performance gap between the optimal policy and the Always Skip policy in the simulation results. Bounding it theoretically is one of our future steps.

4.5.2 Updates of Non-uniform Sizes

In this subsection, we focus on the non-uniform update size case. For illustration, we consider a scenario where the updates are of two possible sizes, and the corresponding transmission times are 5 and 8 time slots, respectively. We assume f_d is a uniform distribution over $\{5, 8\}$, and the probability of arrival $p = 0.14$.

We first obtain the the optimal policy according to Algorithm 10. The policy

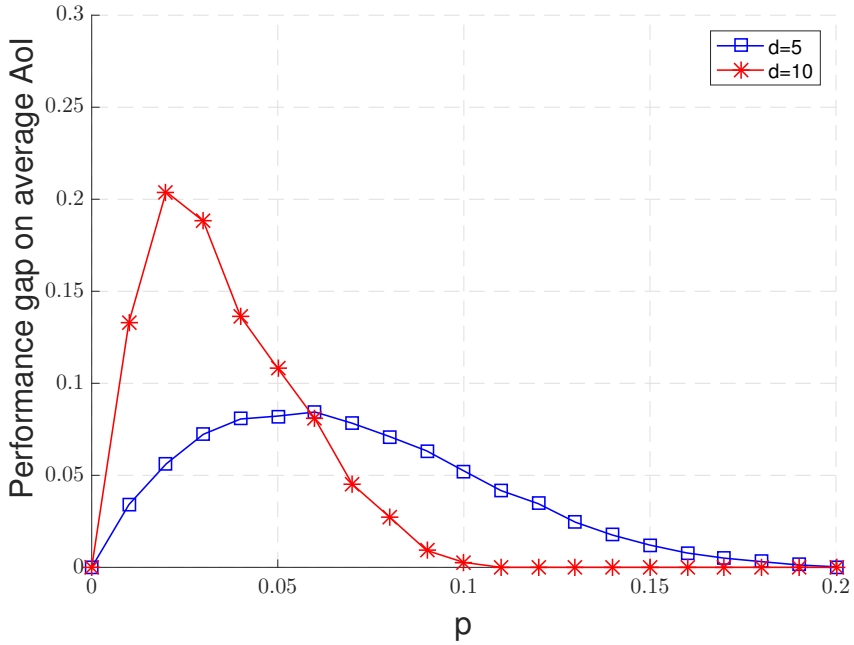
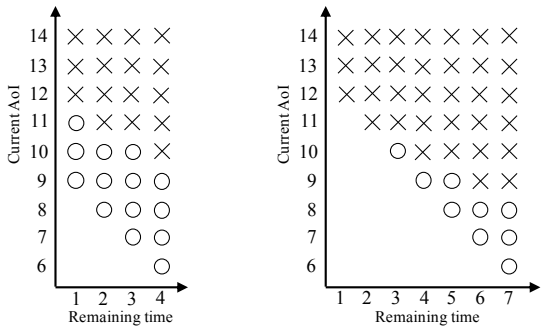


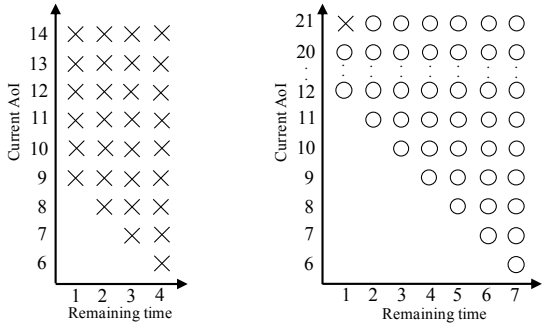
Figure 4.6: Performance gap between the optimal multiple-threshold policy and a myopic policy.

depends on the instantaneous state $\mathbf{s} := (\delta, l, c, b)$, i.e., the current AoI δ, the remaining transmission time of the active update l, the sizes of the active update and the new update, c and b, respectively. Thus, we plot the optimal policy for each fixed (c, b) pair in Fig. 4.7. We note that the optimal policy exhibits the structural properties predicted in Theorem 6. We also note that when $c = 5, b = 8$, the source always prefers to skip the new update. This can be explained by the intuition that switching to an update with longer transmission time will lead to larger AoI when p is not very small.

Then, we evaluate the time-average AoI performances under the optimal policy identified by Algorithm 10, the Always Skip policy, and the Always Switch policy



(a) $c = 5$, $b = 5$. (b) $c = 8$, $b = 8$.



(c) $c = 5$, $b = 8$. (d) $c = 8$, $b = 5$.

Figure 4.7: The optimal policy when $p = 0.14$. Circles represent *switch*, while crosses represent *skip*.

for different p. We note the optimal policy outperforms the other two policies. Compared with the uniform update size case, the performance gap between the optimal policy and the Always Skip policy is more significant. This indicates that switching to a new update of small size would make a substantial impact on the overall AoI.

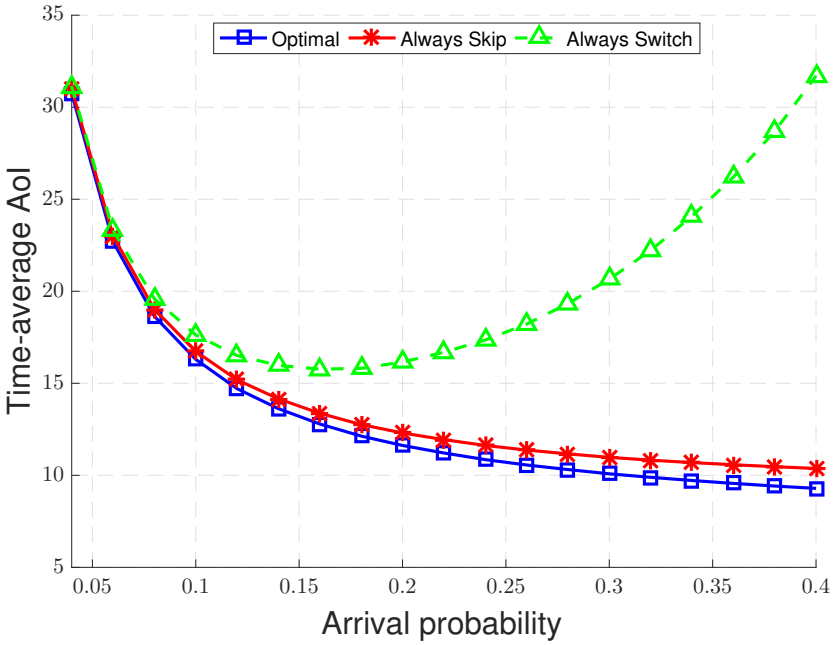


Figure 4.8: Average AoI comparison. $b \in \{5, 8\}$.

4.6 Conclusions

In this paper, we considered a single-link status updating system under link capacity constraint. We first assumed the update size is uniform, and proved that within a broadly defined class of online policies, the optimal policy should be a renewal policy, and has a sequential switching property. We then showed that the optimal decision of the source in any time slot has a multiple-threshold structure, and only depends on the age of the update being transmitted and the AoI in the system. We then considered a more general case that the updates have different sizes, and showed the optimal Markovian policy exhibits certain threshold structures along different dimensions of the system state. For both cases, the optimal policies are numerically

identified through structured value iteration under the MDP framework.

Appendix A

Proofs for Chapter 3

A.1 Proof of Theorem 1

We prove it through induction. Denote the optimal solution as $\{\hat{x}_n\}$, and the output of Algorithm 3 as $\{x_n\}$. We first show that $\hat{x}_{I_1}$ must equal x_{I_1}. We will then show that if $\hat{x}_n = x_n$ for $n = I_1, \ldots, I_k$, we must have $\hat{x}_{I_{k+1}} = x_{I_{k+1}}$.

We assume the optimal schedule is $\{\hat{x}^i_n\}$. We note that x_{I_1} is the maximum energy that can be charged at TS I_1 without violating constraints (3.5), (3.7) and (3.8). We have two possible cases.

Case 1: $s_i = d_i = I_1$ for all $f_i \in \mathcal{F}_{\mathcal{A}}(I_1)$. For this case, $x_{I_1} = \sum_{i:f_i \in \mathcal{F}_{\mathcal{A}}(I_1)} e_i$ under the assumption that $\mathcal{F}_{\mathcal{A}}$ is feasible. Then, we must have $\hat{x}_{I_1} = x_{I_1}$, $\hat{x}^i_{I_1} = e_i$.

Case 2: There exists at least one active demand at I_1 whose span exceeds I_1. If $\hat{x}_{I_1} < x_{I_1}$, then, there must exists a flow i with $\hat{x}^i_{I_1} < x^i_{I_1}$. Besides, there must exist $n \in f_i$, with $\hat{x}^i_{I_n} > x^i_{I_n}$, otherwise, we can always get more profit by

another feasible schedule with $\tilde{x}^i_{I_1} = \hat{x}^i_{I_1} + \varepsilon$, $\tilde{x}^i_n = \hat{x}^i_n - \varepsilon$, since $R_{I_1} > R_n$, where $0 < \varepsilon \leq \min(x_{I_1} - \hat{x}_{I_1}, \hat{x}_n)$.

Thus for all two cases, at the TS I_1, the optimal total charging energy $\hat{x}_{I_1}$ must be equal to x_{I_1}.

Then, assume the total charged energy at TSs $\{I_1, \cdots I_k\}$ that can be charged at TS I_{k+1} without violating constraints (3.5), (3.7) and (3.8). Partition the time axis into non-overlapping blocks, such that each demand belongs to one of them. Denote the block that I_{k+1} belongs to as $\mathcal{B}_{I_{k+1}}$. Then, we have two possible cases.

Case 1: $\mathcal{B}_{I_{k+1}} \subseteq \{I_1, \cdots, I_{k+1}\}$, i.e., $\{x_n\}_{n \in \mathcal{B}_{I_{k+1}}}$ has been determined except $x_{I_{k+1}}$. Then, to maximize the profit, we must have $\hat{x}_{I_{k+1}} = x_{I_{k+1}}$.

Case 2: $\mathcal{B}_{I_{k+1}} \cap \{I_{k+2}, \cdots, I_N\} \neq \emptyset$, i.e., there exists at least one TS in $\mathcal{B}_{I_{k+1}}$ other than I_{k+1} over which the total charging amount is not determined yet. If $\hat{x}_{I_{k+1}} \neq x_{I_{k+1}}$, there exist two possible scenarios:

Case 2(a): There exists a demand $f_i \in \mathcal{F}_{\mathcal{A}}(I_{k+1})$, and a TS $m \in \mathcal{B}_{I_{k+1}} \setminus \{I_1, \cdots, I_{k+1}\}$, such that $\hat{x}^i_{I_{k+1}} < x_{I_{k+1}}$, and $\hat{x}^i_m > x^i_m$. Since $R_m \leq R_{I_{k+1}}$, following similar argument as for I_1, we can show that by letting $\tilde{x}^i_{I_{k+1}} = \hat{x}^i_{I_{k+1}} + \varepsilon$, $\tilde{x}^i_m = \hat{x}^i_m - \varepsilon$ with a sufficiently small ε, the profit won't decrease.

Case 2(b): There exist two demands $f_i \in \mathcal{F}_{\mathcal{A}}(I_{k+1})$, and f_j whose duration is contained in $\mathcal{B}_{I_{k+1}}$, and two TSs $m \in \mathcal{B}_{I_{k+1}} \setminus \{I_1, \cdots, I_{k+1}\}$ and $n \in \mathcal{B}_{I_{k+1}} \cap \{I_1, \cdots, I_k\}$ such that $\hat{x}^i_{I_{k+1}} < x^i_{I_{k+1}}$, $\hat{x}^i_n > x^i_n$, $\hat{x}^j_n < x^j_n$, $\hat{x}^j_m > x^j_m$. Then, we can always let $\tilde{x}^i_{I_{k+1}} = \hat{x}^i_{I_{k+1}} + \varepsilon$, $\tilde{x}^i_n = \hat{x}^i_n - \varepsilon$, $\tilde{x}^j_n = \hat{x}^j_n + \varepsilon$, $\tilde{x}^j_m = \hat{x}^j_m - \varepsilon$. When ε is

sufficiently small, the new allocation is feasible and won't decrease the profit.

Thus, for both cases, if $\hat{x}^i_{I_{k+1}} \neq x^i_{I_{k+1}}$ for $i \in \mathcal{F}_{\mathcal{A}}(I_{k+1})$, we can always increase $\hat{x}^i_{I_{k+1}}$ without compromising the optimality. Therefore, $\hat{x}_{I_{k+1}} = x_{I_{k+1}}$.

In summary, we must have $\hat{x}_n = x_n$ for every n.

A.2 Proof of Theorem 2

We prove this by reducing the well known NP-complete subset sum problem [120] to (4.1). Given a set $N = \{1, 2, \ldots, n\}$ of n items with positive integer weights $\{w_i\}_{i=1}^n$ and a total weight capacity c, the subset sum problem is to find the maximum total weight under the capacity c. We construct n demands $f_1, f_2, \ldots, f_n$ with $s_i = 1, d_i = 1, e_i = w_i$, $i = 1, 2, \ldots, n$. Further, we set $p_{max} = c/\tau$, and $R_1 = 1$. The subset sum problem can be solved if and only if we can find the optimal admission control to maximize the revenue under the maximum charging power constraint p_{max}. The admitted demands represent the subset we are trying to find in the subset sum problem. Therefore the subset sum problem can be reduced in polynomial time to our problem, which proves the NP completeness of the admission control problem.

Appendix B

Proofs for Chapter 4

B.1 Proof of Lemma 7

The proof of this lemma is adapted from from the proof of Theorem 3 in [97]. For the completeness, we provided the detailed proof here.

Denote $F_n(t)$ as the cumulative distribution function of S_n under a uniform bounded policy, i.e., $F_n(t) = \mathbb{P}[S_n \leq t]$. Recall that $N(t)$ is the number of successfully delivered status updates over $(0, t]$. We have

$$\mathbb{E}[N(t)] = \sum_{n=0}^{\infty} F_n(t). \tag{B.1}$$

We note that

$$\mathbb{E}[X_{n+1}^2 \mathbf{1}_{S_{n+1}>T} | S_n = t]$$

$$= \mathbb{E}[X_{n+1}^2 \mathbf{1}_{X_{n+1}>T-t} | S_n = t] \tag{B.2}$$

$$\leq \mathbb{E}_{\mathbf{x}}[g^2(\mathbf{x}) \mathbf{1}_{g(\mathbf{x})>T-t} | S_n = t, \mathbf{x}_n = \mathbf{x}] \tag{B.3}$$

$$= \mathbb{E}_{\mathbf{x}}[g^2(\mathbf{x}) \mathbf{1}_{g(\mathbf{x})>T-t} | \mathbf{x}_n = \mathbf{x}] \tag{B.4}$$

$$:= G(T-t), \tag{B.5}$$

where (B.3) follows from the definition of uniformly bounded policy and (B.4) follows from the fact that $g(\mathbf{x})$ is independent of other parameters. We have the following observation:

$$\lim_{\Delta \to \infty} G(\Delta) = 0. \tag{B.6}$$

We have

$$\mathbb{E}[X_{N(T)+1}^2]$$

$$= \sum_{n=0}^{\infty} \int_0^T \mathbb{E}[X_{n+1}^2 \mathbf{1}_{S_{n+1}>T} | S_n = t] dF_n(t) \tag{B.7}$$

$$\leq \int_0^T G(T-t) d\left(\sum_{n=0}^{\infty} F_n(t) \right) \tag{B.8}$$

$$= \int_0^T G(T-t) d\mathbb{E}[N(t)], \tag{B.9}$$

where (B.8) follows from (B.5), and (B.9) follows from (B.1).

For any fixed Δ satisfying $0 \leq \Delta \leq T$, we have

$$\begin{aligned}
&\frac{1}{T}\int_0^T G(T-t)d\mathbb{E}[N(t)] \\
&= \frac{1}{T}\int_0^{T-\Delta} G(T-t)d\mathbb{E}[N(t)] + \frac{1}{T}\int_{T-\Delta}^{T} G(T-t)d\mathbb{E}[N(t)] \qquad \text{(B.10)} \\
&\leq G(\Delta)\frac{\mathbb{E}[N(T-\Delta)]}{T} + G(0)\frac{\mathbb{E}[N(T)] - \mathbb{E}[N(T-\Delta)]}{T} \qquad \text{(B.11)} \\
&\leq G(\Delta)\frac{T-\Delta}{dT} + G(0)\frac{\Delta}{dT}, \qquad \text{(B.12)}
\end{aligned}$$

where (B.11) follows from that fact that $G(t)$ is a non-increasing function, (B.11) follows from the fact that time elapsed between two successful updates is greater than or equal to the d time slots.

Therefore,

$$\begin{aligned}
&\lim_{T\to\infty} \frac{1}{T}\int_0^T G(T-t)d\mathbb{E}[N(T)] \\
&= \frac{G(\Delta)}{d}, \qquad \text{(B.13)}
\end{aligned}$$

where (B.13) holds for any $\Delta \geq 0$. Since $\lim_{\Delta\to\infty} G(\Delta) = 0$ in (B.6), we have $\lim_{T\to\infty} \frac{\mathbb{E}\left[X^2_{N(T)+1}\right]}{T} = 0$.

B.2 Proof of Theorem 3

Denote $\{X_i\}$ as the inter-update delays generated under a uniformly bounded policy π. For the same history $\mathcal{H}^{i-1}$, all the status updating sample paths that share the

same update arrival patten $\boldsymbol{\tau}_i$, perform a statistical averaging over all of them to get the following average age in the ith epoch

$$\hat{R}_i(\gamma, \mathcal{H}^{i-1}) := \mathbb{E}[R_i|\boldsymbol{\tau}_i = \gamma, \mathcal{H}^{i-1}], \tag{B.14}$$

where γ is an update arrival patten over the i^{th} updating epoch. Then, we have

$$\mathbb{E}[R_i \mathbf{1}_{i\leq N(T)+1}] = \mathbb{E}_{\mathcal{H}^{i-1}}[\mathbb{E}_\gamma[\hat{R}_i(\gamma, \mathcal{H}^{i-1})]\mathbf{1}_{i\leq N(T)+1}|\mathcal{H}^{i-1}], \tag{B.15}$$

where the equality follows from the fact that $\mathbf{1}_{i\leq N(T)+1}$ is independent of $\boldsymbol{\tau}_i$ given $\mathcal{H}^{i-1}$.

Similarly, define the average ith epoch length as

$$\hat{X}_i(\gamma, \mathcal{H}^{i-1}) := \mathbb{E}[X_i|\boldsymbol{\tau}_i = \gamma, \mathcal{H}^{i-1}]. \tag{B.16}$$

According to Lemma 7, we have $\lim_{T\to\infty} \frac{X^2_{N(T)+1}}{T} = 0$. Then,

$$\lim_{T\to\infty} \frac{\mathbb{E}[R(T)]}{T} = \lim_{T\to\infty} \frac{\mathbb{E}\left[\sum_{i=1}^{N(T)+1} R_i\right]}{T} \tag{B.17}$$

$$\geq \frac{\mathbb{E}\left[\sum_{i=1}^{N(T)+1} R_i\right]}{2\mathbb{E}[S_{N(T)+1}]} = \frac{\sum_{i=1}^{\infty} \mathbb{E}\left[R_i \mathbf{1}_{i\leq N(T)+1}\right]}{\sum_{i=1}^{\infty} \mathbb{E}[X_i \mathbf{1}_{i\leq N(T)+1}]} \tag{B.18}$$

$$= \frac{\sum_{i=1}^{\infty} \mathbb{E}_{\mathcal{H}^{i-1}}\left[\mathbb{E}_\gamma\left[\hat{R}_i(\gamma, \mathcal{H}^{i-1})\right] \mathbf{1}_{i\leq N(T)+1}|\mathcal{H}^{i-1}\right]}{\sum_{i=1}^{\infty} \mathbb{E}[X_i \mathbf{1}_{i\leq N(T)+1}]} \tag{B.19}$$

$$= \frac{\sum_{i=1}^{\infty} \mathbb{E}_{\mathcal{H}^{i-1}}\left[\mathbb{E}_\gamma[\hat{X}_i(\gamma, \mathcal{H}^{i-1})] \frac{\mathbb{E}_\gamma\left[\hat{R}_i(\gamma, \mathcal{H}^{i-1})\right]\mathbf{1}_{i\leq N(T)+1}}{\mathbb{E}_\gamma[\hat{X}_i(\gamma, \mathcal{H}^{i-1})]}|\mathcal{H}^{i-1}\right]}{\sum_{i=1}^{\infty} \mathbb{E}[X_i \mathbf{1}_{i\leq N(T)+1}]} \tag{B.20}$$

$$\geq \frac{\sum_{i=1}^{\infty} \mathbb{E}_{\mathcal{H}^{i-1}} \left[\mathbb{E}_{\gamma}[\hat{X}_i(\gamma, \mathcal{H}^{i-1})] R^* \mathbf{1}_{i \leq N(T)+1} | \mathcal{H}^{i-1} \right]}{\sum_{i=1}^{\infty} \mathbb{E}[X_i \mathbf{1}_{i \leq N(T)+1}]} \tag{B.21}$$

$$= R^* \tag{B.22}$$

where $R^* := \min_{i, \mathcal{H}^{i-1}} \frac{\mathbb{E}_{\gamma}\left[\hat{R}_i(\gamma, \mathcal{H}^{i-1})\right]}{\mathbb{E}_{\gamma}[\hat{X}_i(\gamma, \mathcal{H}^{i-1})]}$, which corresponds to the expected average AoI achieved by a renewal policy under which X_i only depends on γ.

B.3 Proof of Lemma 9

We prove this lemma through contradiction. Now assume the optimal policy π_0 is not an SS policy. Without loss of generality, we consider the first renewal epoch starting at time 0 (the beginning of time slot 1). We assume under π_0 there exists a sample path under which the source transmits the new update arrival at time slot i and does not switch to the next arrival at time slot j in the same epoch, i.e., $i < j < i + d$. Depending on the upcoming random arrivals, the sample path may evolve into different sample paths. Denote the set of such sample paths as $\mathcal{F}_j$, as they share the same history up to time slot j. We can partition $\mathcal{F}_j$ into two subsets:

- $\mathcal{F}_{j,1}$: The source skips all the upcoming arrivals and finishes transmitting the update arrives at i.
- $\mathcal{F}_{j,2}$: The source switches to some later arrival.

Let X^{π_0} be the corresponding length of the renewal epoch under policy π_0. Then,

$X^{\pi_0} = i + d - 1$ for sample paths in $\mathcal{F}_{j,1}$, and $X^{\pi_0} > j + d - 1$ for sample paths in $\mathcal{F}_{j,2}$.

We now construct two policies π_1 and π_2 as follows. Under both π_1 and π_2, the source will behave exactly the same as under π_0 for all sample paths not in $\mathcal{F}_j$. However, for the sample paths in $\mathcal{F}_j$, the actions the source will take after j will be different. Specifically, under π_1, the source will finish the update that arrives at time slot i irrespective of other factors. Therefore, for all sample paths in $\mathcal{F}_j$ under π_0, the corresponding length of the renewal epoch under π_1 will be $X^{\pi_1} = i + d - 1$ under π_1. For π_2, we will let the source first switch to the arrival at time slot j, and then switch to a later arrival whenever the source switches under π_0. Then, for the sample paths in $\mathcal{F}_{j,1}$ under π_0, the corresponding length of renewal epoch will be changed to $X^{\pi_2} = j + d - 1$ under π_2; while for those in $\mathcal{F}_{j,2}$, $X^{\pi_2} = X^{\pi_0}$.

Therefore, considering all possible sample paths under those policies, we have $\mathbb{E}[X^{\pi_1}] < \mathbb{E}[X^{\pi_0}] < \mathbb{E}[X^{\pi_2}]$, which implies that there must exist a ρ, $0 < \rho < 1$, such that

$$\rho\mathbb{E}[X^{\pi_1}] + (1 - \rho)\mathbb{E}[X^{\pi_2}] = \mathbb{E}[X^{\pi_0}]. \tag{B.23}$$

We will then construct a randomized policy π', under which it follows π_1 with probability ρ and follows π_2 with probability $1 - \rho$. Apparently, the expected length of the renewal epoch under π', denoted as $X^{\pi'}$, will be the same as that under π_0.

Next, we will show that $\mathbb{E}[(X^{\pi'})^2] \leq \mathbb{E}[(X^{\pi_0})^2]$. Denote $P_1 := \frac{\mathbb{P}_{\pi_0}[\mathcal{F}_{j,1}]}{\mathbb{P}_{\pi_0}[\mathcal{F}_j]}, P_2 :=$

$\frac{\mathbb{P}_{\pi_0}[\mathcal{F}_{j,2}]}{\mathbb{P}_{\pi_0}[\mathcal{F}_j]}$. Then, (B.23) can be expressed as

$$\begin{aligned}&\rho(i+d-1)+(1-\rho)\left[(j+d-1)P_1+\mathbb{E}[X^{\pi_0}|\mathcal{F}_{j,2}]P_2\right]\\&=(i+d-1)P_1+\mathbb{E}[X^{\pi_0}|\mathcal{F}_{j,2}]P_2,\end{aligned}\tag{B.24}$$

which can be reduced to

$$\begin{aligned}&(1-\rho)P_1(j+d-1)\\&=(P_1-\rho)(i+d-1)+\rho P_2\mathbb{E}[X^{\pi_0}|\mathcal{F}_{j,2}].\end{aligned}\tag{B.25}$$

Since $\mathbb{E}[X^{\pi_0}|\mathcal{F}_{j,2}] > j+d-1$, $(1-\rho)P_1 = (P_1-\rho)+\rho P_2$, (B.25) implies that $P_1-\rho>0$. Dividing both sides of (B.25) by $(1-\rho)P_1$, we have

$$j+d-1=\frac{P_1-\rho}{(1-\rho)P_1}(i+d-1)+\frac{\rho P_2}{(1-\rho)P_1}\mathbb{E}[X^{\pi_0}|\mathcal{F}_{j,2}].$$

Note that $\frac{P_1-\rho}{(1-\rho)P_1}$ and $\frac{\rho P_2}{(1-\rho)P_1}$ form a valid distribution. Therefore, based on Jensen's inequality, we have

$$\begin{aligned}(j+d-1)^2&<\frac{P_1-\rho}{(1-\rho)P_1}(i+d-1)^2+\frac{\rho P_2}{(1-\rho)P_1}\left(\mathbb{E}[X^{\pi_0}|\mathcal{F}_{j,2}]\right)^2\\&\leq\frac{P_1-\rho}{(1-\rho)P_1}(i+d-1)^2+\frac{\rho P_2}{(1-\rho)P_1}\mathbb{E}[(X^{\pi_0})^2|\mathcal{F}_{j,2}],\end{aligned}\tag{B.26}$$

which is equivalently to

$$\rho(i+d-1)^2 + (1-\rho)\left[(j+d-1)^2 P_1 + \mathbb{E}[(X^{\pi_0})^2 \,|\mathcal{F}_{j,2}]P_2\right]$$
$$< (i+d-1)^2 P_1 + \mathbb{E}[(X^{\pi_0})^2 | \mathcal{F}_{j,2}]P_2. \tag{B.27}$$

I.e.,

$$\rho\mathbb{E}[(X^{\pi_1})^2] + (1-\rho)\mathbb{E}[(X^{\pi_2})^2] < \mathbb{E}[(X^{\pi_0})^2]. \tag{B.28}$$

Combining (B.23) and (B.28), we have

$$\frac{1}{2}\frac{\rho\mathbb{E}[(X^{\pi_1})^2] + (1-\rho)\mathbb{E}[(X^{\pi_2})^2]}{\rho\mathbb{E}[X^{\pi_1}] + (1-\rho)\mathbb{E}[X^{\pi_2}]} < \frac{1}{2}\frac{\mathbb{E}[(X^{\pi_0})^2]}{\mathbb{E}[X^{\pi_0}]}, \tag{B.29}$$

i.e., the new policy π' achieves a lower expected average AoI than π_0, which contradicts with the assumption that π_0 is optimal.

B.4 Proof of Lemma 10

We prove this lemma through contradiction. Again, we focus on the first renewal epoch that starts at time 0. Assume under the optimal SS policy π_0, the transmitter transmits the update that arrives in time slot i. It will then skip the next arrival if it arrives at time slot j, and switches to it if it arrives at time slot k, with $j < k < i + d$. We aim to perturb π_0 a bit to obtain a new policy π' and show that

it achieves a lower expected average AoI.

Similarly to the proof of Lemma 9, we consider all sample paths under π_0 that choose to transmit the update that arrives at time slot i, while the next arrival time is either j or k. Denote the subset of such sample paths as $\mathcal{F}_i$. Again, we divide $\mathcal{F}_i$ into two subsets.

- $\mathcal{F}_{i,1}$: The next arrival time is j. Then, the sample paths will skip j and finish transmitting i. The corresponding length of renewal interval $X^{\pi_0} = i + d - 1$.
- $\mathcal{F}_{i,2}$: The next arrival time is k. The sample paths will switch to the new arrival, and the corresponding length of renewal interval $X^{\pi_0} \geq k + d - 1$.

Then, we construct the new policy π' in this way: For all sample paths in $\mathcal{F}_{i,1}$, the new policy will let the transmitter randomly switch to the next arrival at j and finish transmitting it with probability ρ_j, and let the rest sample paths follow the original policy π_0, i.e., skip the new arrival and finish transmitting the arrival at i. For all sample paths in $\mathcal{F}_{i,2}$, the new policy will let the transmitter randomly skip the next arrival at k with probability ρ_k, and let the rest sample paths follow the original policy π_0. Then, under π', we have

$$\mathbb{E}[X^{\pi'}|\mathcal{F}_{i,1}] = \rho_j(j + d - 1) + (1 - \rho_j)(i + d - 1), \tag{B.30}$$

$$\mathbb{E}[X^{\pi'}|\mathcal{F}_{i,2}] = \rho_k(i + d - 1) + (1 - \rho_k)\mathbb{E}[X^{\pi_0}|\mathcal{F}_{i,2}]. \tag{B.31}$$

Since $i + d + 1 < j + d + 1 < k + d + 1$, there must exist a pair of ρ_j and ρ_k such

that

$$\mathbb{E}[X^{\pi'}|\mathcal{F}_i] = \mathbb{E}[X^{\pi_0}|\mathcal{F}_i], \tag{B.32}$$

i.e.,

$$\rho_j(j-i)\mathbb{P}[\mathcal{F}_{i,1}] = \rho_k[\mathbb{E}[X^{\pi_0}|\mathcal{F}_{i,2}] - (i+d-1)]\mathbb{P}[\mathcal{F}_{i,2}]. \tag{B.33}$$

Following similar argument as in the proof of Lemma 9, we can show that

$$\mathbb{E}[(X^{\pi'})^2|\mathcal{F}_i] \leq \mathbb{E}[(X^{\pi_0})^2|\mathcal{F}_i] \tag{B.34}$$

Separating term $(j+d-1)$ from equation (B.33), we have

$$j+d-1 = \frac{\rho_j\mathbb{P}[\mathcal{F}_{i,1}]-\rho_k\mathbb{P}[\mathcal{F}_{i,2}]}{\rho_j\mathbb{P}[\mathcal{F}_{i,1}]}(i+d-1) + \frac{\rho_k\mathbb{P}[\mathcal{F}_{i,2}]}{\rho_j\mathbb{P}[\mathcal{F}_{i,1}]}\mathbb{E}[X^{\pi_0}|\mathcal{F}_{i,2}] \tag{B.35}$$

Since $\mathbb{E}[X^{\pi_0}|\mathcal{F}_{i,2}] \geq k+d-1$ and $i<j<k$, from equation (B.33) we must have

$$\frac{\rho_k\mathbb{P}[\mathcal{F}_{i,2}]}{\rho_j\mathbb{P}[\mathcal{F}_{i,1}]} = \frac{j-i}{\mathbb{E}[X^{\pi_0}|\mathcal{F}_{i,1}]-(i+d-1)} \leq \frac{j-i}{k-i} \leq 1 \tag{B.36}$$

Thus $\frac{\rho_j\mathbb{P}[\mathcal{F}_{i,1}]-\rho_k\mathbb{P}[\mathcal{F}_{i,2}]}{\rho_j\mathbb{P}[\mathcal{F}_{i,1}]}$ and $\frac{\rho_k\mathbb{P}[\mathcal{F}_{i,2}]}{\rho_j\mathbb{P}[\mathcal{F}_{i,1}]}$ form a valid distribution.

From Jensen's inequality, we have

$$(j+d-1)^2 \leq \frac{\rho_j \mathbb{P}[\mathcal{F}_{i,1}] - \rho_k \mathbb{P}[\mathcal{F}_{i,2}]}{\rho_j \mathbb{P}[\mathcal{F}_{i,1}]}(i+d-1)^2 + \frac{\rho_k \mathbb{P}[\mathcal{F}_{i,2}]}{\rho_j \mathbb{P}[\mathcal{F}_{i,1}]}\left(\mathbb{E}[X^{\pi_0}|\mathcal{F}_{i,2}]\right)^2 \tag{B.37}$$

$$\leq \frac{\rho_j \mathbb{P}[\mathcal{F}_{i,1}] - \rho_k \mathbb{P}[\mathcal{F}_{i,2}]}{\rho_j \mathbb{P}[\mathcal{F}_{i,1}]}(i+d-1)^2 + \frac{\rho_k \mathbb{P}[\mathcal{F}_{i,2}]}{\rho_j \mathbb{P}[\mathcal{F}_{i,1}]}\mathbb{E}[(X^{\pi_0})^2|\mathcal{F}_{i,2}] \tag{B.38}$$

Thus we have

$$\mathbb{E}[(X^{\pi'})^2|\mathcal{F}_i] = \mathbb{P}[\mathcal{F}_{i,1}]\left(\rho_j(j+d-1)^2 + (1-\rho_j)(i+d-1)^2\right) + \mathbb{P}[\mathcal{F}_{i,2}]\left(\rho_k(i+d-1)^2 + (1-\rho_k)\mathbb{E}[(X^{\pi_0})^2|\mathcal{F}_{i,2}]\right) \tag{B.39}$$

$$\leq \mathbb{P}[\mathcal{F}_{i,1}](i+d-1)^2 + \mathbb{P}[\mathcal{F}_{i,2}]\mathbb{E}[(X^{\pi_0})^2|\mathcal{F}_{i,2}] \tag{B.40}$$

$$= \mathbb{E}[(X^{\pi_0})^2|\mathcal{F}_i]. \tag{B.41}$$

Thus policy π' achieves lower expected average AoI which follows from (B.32), which contradicts with the assumption that π_0 is optimal. Therefore, if the source skips the next update if it arrives at time slot j, it must skip it if it arrives later than j. Picking the minimum j ($j > i$) in which a skip would occur, denote it as j^*. Then, the threshold τ_i equals $j^* - 1$. Since the proof does not depend other information except i, τ_i depends on i only.

B.5 Proof of Theorem 4

Recall that we assume $d \geq 2$, if $d = 2$, we have only one threshold τ_2, thus we consider $d > 2$ with three possible scenarios.

1) $\tau_i > i + 1$.

We aim to show that the switch thresholds has monotonicity property, i.e., $\tau_{i+1} \leq \tau_i$.

We prove it through contradiction. Assume under optimal policy π_0 there exists two subsets of sample paths as follows.

I) $\mathcal{F}_{i,\tau_i+1}$: The transmitter transmits the arrival at i and the next arrival is at $\tau_i + 1$. Then, under π_0, it will skip the arrival and finish transmitting the arrival at i. The corresponding $X^{\pi_0} = i + d - 1$.

II) $\mathcal{F}_{i+1,\tau_i+1}$: The transmitter transmits the arrival at $i + 1$ and the next arrival is at $\tau_i + 1$. Since $\tau_{i+1} > \tau_i$, under π_0, it will switch to the new arrival. The corresponding $X^{\pi_0} \geq \tau_i + d$.

Then, we will modify π_0 to π' as follows: For sample paths in $\mathcal{F}_{i,\tau_i+1}$, the transmitter will randomly switch to the new arrival at $\tau_i + 1$ with probability ρ_i and then follow policy π_0 afterwards. Note that the sample paths after switching at $\tau_i + 1$ should be the same as those in $\mathcal{F}_{i+1,\tau_i+1}$. Then, we have

$$\mathbb{E}[X^{\pi'}|\mathcal{F}_{i,\tau_i+1}] = \rho_i \mathbb{E}[X^{\pi_0}|\mathcal{F}_{i+1,\tau_i+1}] + (1 - \rho_i)(i + d - 1). \tag{B.42}$$

For sample paths in $\mathcal{F}_{i+1,\tau_i+1}$, the transmitter will randomly skip the new arrival at τ_i+1 with probability ρ_{i+1} and finish transmitting the arrival at $i+1$; It will stick with policy π_0 for the rest sample paths. Then

$$\mathbb{E}[X^{\pi'}|\mathcal{F}_{i+1,\tau_i+1}] = \rho_{i+1}(i+d) + (1-\rho_{i+1})\mathbb{E}[X^{\pi_0}|\mathcal{F}_{i+1,\tau_i+1}]. \tag{B.43}$$

Note that $0 \leq \rho_i, \rho_{i+1} \leq 1$, we have

$$(i+d-1)P_I + (i+d)P_{II} \leq \mathbb{E}[X^{\pi'}|\mathcal{F}_{i,\tau_i+1}]P_I + \mathbb{E}[X^{\pi'}|\mathcal{F}_{i+1,\tau_i+1}]P_{II} \leq \mathbb{E}[X^{\pi_0}|\mathcal{F}_{i+1,\tau_i+1}]. \tag{B.44}$$

We also have

$$\mathbb{E}[X^{\pi_0}|\mathcal{F}_{i,\tau_i+1}]P_I + \mathbb{E}[X^{\pi_0}|\mathcal{F}_{i+1,\tau_i+1}]P_{II} = (i+d-1)P_I + \mathbb{E}[X^{\pi_0}|\mathcal{F}_{i+1,\tau_i+1}]P_{II}. \tag{B.45}$$

Thus we can pick ρ_i and ρ_{i+1} in a way such that

$$\mathbb{E}[X^{\pi'}|\mathcal{F}_{i,\tau_i+1}]P_I + \mathbb{E}[X^{\pi'}|\mathcal{F}_{i+1,\tau_i+1}]P_{II} = \mathbb{E}[X^{\pi_0}|\mathcal{F}_{i,\tau_i+1}]P_I + \mathbb{E}[X^{\pi_0}|\mathcal{F}_{i+1,\tau_i+1}]P_{II}, \tag{B.46}$$

where $P_I := \frac{\mathbb{P}[\mathcal{F}_{i,\tau_i+1}]}{\mathbb{P}[\mathcal{F}_{i,\tau_i+1}]+\mathbb{P}[\mathcal{F}_{i+1,\tau_i+1}]}$, $P_{II} := \frac{\mathbb{P}[\mathcal{F}_{i+1,\tau_i+1}]}{\mathbb{P}[\mathcal{F}_{i,\tau_i+1}]+\mathbb{P}[\mathcal{F}_{i+1,\tau_i+1}]}$. Equation (B.46) implies

that

$$\rho_{i+1}P_{II}(i+d) = (\rho_{i+1}P_{II} - \rho_i P_I)\mathbb{E}[X^{\pi_0}|\mathcal{F}_{i+1,\tau_i+1}] + \rho_i P_I(i+d-1). \tag{B.47}$$

Since $\mathbb{E}[X^{\pi_0}|\mathcal{F}_{i+1,\tau_i+1}] \geq \tau_i + d > i + d$, from equation (B.47) we must have

$$\frac{\rho_i P_I}{\rho_{i+1}P_{II}} = \frac{\mathbb{E}[X^{\pi_0}|\mathcal{F}_{i+1,\tau_i+1}] - (i+d)}{\mathbb{E}[X^{\pi_0}|\mathcal{F}_{i+1,\tau_i+1}] - (i+d-1)} < 1. \tag{B.48}$$

Thus $\frac{\rho_{i+1}P_{II}-\rho_i P_I}{\rho_{i+1}P_{II}}$ and $\frac{\rho_i P_I}{\rho_{i+1}P_{II}}$ form a valid distribution. Therefore, based on Jensen's inequality, we have

$$(i+d)^2 < \frac{\rho_{i+1}P_{II} - \rho_i P_I}{\rho_{i+1}P_{II}}(\mathbb{E}[X^{\pi_0}|\mathcal{F}_{i+1,\tau_i+1}])^2 + \frac{\rho_i P_I}{\rho_{i+1}P_{II}}(i+d-1)^2 \tag{B.49}$$

$$\leq \frac{\rho_{i+1}P_{II} - \rho_i P_I}{\rho_{i+1}P_{II}}\mathbb{E}[(X^{\pi_0})^2|\mathcal{F}_{i+1,\tau_i+1}] + \frac{\rho_i P_I}{\rho_{i+1}P_{II}}(i+d-1)^2, \tag{B.50}$$

which implies that

$$\rho_{i+1}P_{II}(i+d)^2 \leq (\rho_{i+1}P_{II} - \rho_i P_I)\mathbb{E}[(X^{\pi_0})^2|\mathcal{F}_{i+1,\tau_i+1}] + \rho_i P_I(i+d-1)^2. \tag{B.51}$$

Therefore

$$\mathbb{E}[(X^{\pi'})^2 \leq \mathbb{E}[(X^{\pi_0})^2]. \tag{B.52}$$

Thus, π_0 cannot be optimal.

2) $\tau_i = i$, i.e., if the transmitter switches to an update arriving at time slot i, it will skip all upcoming packets until the transmission is completed.

We aim to show that if the transmitter switches to an update arriving at time slot $i+1$, it will also stick with it until it finishes the transmission. We prove this through contradiction, i.e., we assume that under the optimal policy π_0, we have $\tau_{i+1} > i+1$. Consider two subset of sample paths as follows.

I) $\mathcal{F}_{i,i+2}$: The transmitter transmits the arrival at i and the next arrival is at $i+2$. Then, under π_0, it will skip the arrival and finish transmitting the arrival at i. The corresponding $X^{\pi_0} = i + d - 1$.

II) $\mathcal{F}_{i+1,i+2}$: The transmitter transmits the arrival at $i+1$ and the next arrival is at $i+2$. Since $\tau_{i+1} > i+1$, under π_0, it will switch to the new arrival. The corresponding $X^{\pi_0} \geq i + 1 + d$.

Then, we modify π_0 to π' as follows. For sample paths in $\mathcal{F}_{i,i+2}$, the transmitter will randomly switch to the new arrival at $i+2$ with probability ρ_i and then follow policy π_0 afterwards. Note that the sample paths after switching at $i+2$ should be the same as those in $\mathcal{F}_{i+1,i+2}$.

For sample paths in $\mathcal{F}_{i+1,i+2}$, the transmitter will randomly skip the new arrival at $i+2$ with probability ρ_{i+1} and finish transmitting the arrival at $i+1$; Otherwise it will follow policy π_0 for the rest sample paths.

Following the same approach as in the first case, we can pick ρ_i and ρ_{i+1} such that $\mathbb{E}[X^{\pi'}] = \mathbb{E}[X^{\pi_0}]$ and $\mathbb{E}[(X^{\pi'})^2 \leq \mathbb{E}[(X^{\pi_0})^2]$. Thus π_0 cannot be optimal.

3) $\tau_a = i + 1$. We aim to show $\tau_{i+1} = i + 1$. The proof is through contradiction and follows exactly the same steps as in case 2).

Combining three cases, we conclude that the threshold τ_1, $\tau_2 \cdots$ are monotonically decreasing until $\tau_K = K$.

B.6 Proof of Theorem 6

In the following, we prove the structural properties of the optimal policy characterized in Theorem 6 one by one.

Proof of Theorem 6(a). Assume at iteration n, the optimal action for state $\mathbf{s}$ is to switch. It implies that $Q_{n-1}(\mathbf{s}; 1) \leq Q_{n-1}(\mathbf{s}; 0)$. Then, for state $\mathbf{s}' := (\delta, l, c, b')$ with $b' < b$, we have $Q_{n-1}(\mathbf{s}'; 1) \leq Q_{n-1}(\mathbf{s}; 1)$, which is shown in the proof of Lemma 12. Together with the fact that $Q_{n-1}(\mathbf{s}'; 0) = Q_{n-1}(\mathbf{s}; 0)$, the optimal action for state $\mathbf{s}'$ at iteration n is to switch as well. This property holds when $n \to \infty$, thus Theorem 6(a) is proved.

Proof of Theorem 6(b). We first consider the case when $b = l$. We have

$$Q_n(\mathbf{s}; 1) = \delta + \alpha \mathbb{E}[V_n^{\alpha}(\delta + 1, b - 1, b, \bar{b})],$$

$$Q_n(\mathbf{s}; 0) = \delta + \alpha \mathbb{E}[V_n^{\alpha}(\delta + 1, l - 1, c, \bar{b})].$$

Based on the fact $l < c$ and the monotonicity of $V_n^{\alpha}(\mathbf{s})$ in c according to Lemma 12, we must have $Q_n(\mathbf{s}; 1) \leq Q_n(\mathbf{s}; 0)$.

Therefore, when $b = l$, the optimal action is to switch. Then, according to Theorem 6(a), when $b < l$, the optimal action is to switch as well.

Proof of Theorem 6(c). Theorem 6(c) can be proved in a way similar to the proof of Theorem 6(b), and is thus omitted.

Proof of Theorem 6(d). To show the optimal decision is to switch, we need to prove $Q_k(\mathbf{s}; 1) \leq Q_k(\mathbf{s}; 0)$. We note that

$$Q_{n-1}((\delta, 0, 0, b); 0) = \delta + \alpha \mathbb{E}[V^{\alpha}_{n-1}(\delta + 1, 0, 0, \bar{b})], \tag{B.53}$$

$$Q_{n-1}((\delta, 0, 0, b); 1) = \delta + \alpha \mathbb{E}[V^{\alpha}_{n-1}(\delta + 1, b-1, b, \bar{b})]. \tag{B.54}$$

When $n < b$, initially, we have

$$\mathbb{E}[V^{\alpha}_0(\delta+n, b-n, b, \bar{b})] \leq \mathbb{E}[V^{\alpha}_0(\delta+n, 0, 0, \bar{b})]. \tag{B.55}$$

Recursively applying Corollary 15, we have

$$\mathbb{E}[V^{\alpha}_{n-1}(\delta+1, b-1, b, \bar{b})] \leq \mathbb{E}[V^{\alpha}_{n-1}(\delta+1, 0, 0, \bar{b})], \tag{B.56}$$

which indicates $\Delta Q^{1,0}_{n-1}(\mathbf{s}) \leq 0$ according to (B.53)-(B.54).

When $n \geq b$, according to the monotonicity of $V(\mathbf{s})$ characterized in Lemma 12,

$$\mathbb{E}[V^{\alpha}_{n-b}(b, 0, 0, \bar{b})] \leq \mathbb{E}[V^{\alpha}_{n-b}(\delta + b, 0, 0, \bar{b})]. \tag{B.57}$$

Then, based on the definition of $Q(\mathbf{s};0)$ in (4.19), we have

$$Q_{n-b}((\delta+b-1,1,b,.);0) \leq Q_{n-b}((\delta+b-1,0,0,.);0). \tag{B.58}$$

Next, according to Lemma 13,

$$\mathbb{E}[V^{\alpha}_{n-b+1}(\delta+b-1,1,b,\bar{b})] \leq \mathbb{E}[V^{\alpha}_{n-b+1}(\delta+b-1,0,0,\bar{b})]. \tag{B.59}$$

Applying Corollary 15 recursively, we have

$$\mathbb{E}[V^{\alpha}_{n-1}(\delta+1,b-1,b,\bar{b})] \leq \mathbb{E}[V^{\alpha}_{n-1}(\delta+1,0,0,\bar{b})]. \tag{B.60}$$

which indicates $\Delta Q^{1,0}_{n-1}(\mathbf{s}') \leq 0$ according to (B.53)-(B.54).

Thus, the optimal decision at state $(\delta,0,0,b)$ when $b \neq 0$ is always to switch.

Proof of Theorem 6(e). We first point out that if the optimal decision at state $\mathbf{s}$ is to skip, it must have $b > l > 0$ according to Theorem 6(b)(d). Thus, there are two possible cases depending on the value l:

Case 1: $l = 1$. For this case,

$$\begin{aligned}&\Delta Q^{1,0}_{n-1}(\delta,l,c,b)\\ &= \alpha\mathbb{E}[V^{\alpha}_{n-1}(\delta+1,b-1,b,\bar{b})] - \alpha\mathbb{E}[V^{\alpha}_{n-1}(c,0,0,\bar{b})],\end{aligned} \tag{B.61}$$

which is increasing in δ according to Lemma 12. Thus, if $\Delta Q^{1,0}_{n-1}(\delta,l,c,b) \geq 0$, we

must have $\Delta Q_{n-1}^{1,0}(\delta+1,l,c,b) \geq 0$.

Case 2: $l > 1$. For this case, we have

$$Q_{n-1}((\delta,l,c,b);0) = \delta + \alpha\mathbb{E}[V_{n-1}^{\alpha}(\delta+1,l-1,c,\bar{b})], \tag{B.62}$$

$$Q_{n-1}((\delta,l,c,b);1) = \delta + \alpha\mathbb{E}[V_{n-1}^{\alpha}(\delta+1,b-1,b,\bar{b})]. \tag{B.63}$$

When $n < l$, initially, we have

$$Q_0((\delta{+}n,l{-}n{+}1,c,.);0) \leq Q_0((\delta{+}n,b{-}n{+}1,b,.);0). \tag{B.64}$$

Based on Lemma 13, we get

$$\mathbb{E}[V_1^{\alpha}(\delta{+}n,l{-}n{+}1,c,\bar{b})] \leq \mathbb{E}[V_1^{\alpha}(\delta{+}n,b{-}n{+}1,b,\bar{b})]. \tag{B.65}$$

Applying Corollary 14, we have

$$\begin{aligned} &\mathbb{E}[V_2^{\alpha}(\delta+n-1,l-n+2,c,\bar{b})] \\ &\leq \mathbb{E}[V_2^{\alpha}(\delta+n-1,b-n+2,b,\bar{b})]. \end{aligned} \tag{B.66}$$

Recursively applying Corollary 14, it leads to

$$\mathbb{E}[V_{n-1}^{\alpha}(\delta+2,l-1,c,\bar{b})] \leq \mathbb{E}[V_{n-1}^{\alpha}(\delta+2,b-1,b,\bar{b})], \tag{B.67}$$

which indicates $\Delta Q_{n-1}^{1,0}(\mathbf{s}') \geq 0$ according to (B.62)-(B.63).

When $n \geq l$, $\Delta Q_{n-1}^{1,0}(\mathbf{s}) \geq 0$ implies that

$$\mathbb{E}[V_{n-1}^{\alpha}(\delta+1, l-1, c, \bar{b})] \leq \mathbb{E}[V_{n-1}^{\alpha}(\delta+1, b-1, b, \bar{b})]. \tag{B.68}$$

according to (B.62)-(B.63).

Based on Corollary 14, we thus have

$$\mathbb{E}[V_{n-2}^{\alpha}(\delta+2, l-2, c, \bar{b})] \leq \mathbb{E}[V_{n-2}^{\alpha}(\delta+2, b-2, b, \bar{b})]. \tag{B.69}$$

We can further obtain the following inequality by applying Corollary 14 recursively:

$$\begin{aligned} &\mathbb{E}[V_{n-l+1}^{\alpha}(\delta+l-1, 1, c, \bar{b})] \\ &\leq \mathbb{E}[V_{n-l+1}^{\alpha}(\delta+l-1, b-l+1, b, \bar{b})]. \end{aligned} \tag{B.70}$$

Thus, according to Lemma 13, we have

$$\begin{aligned} &Q_{n-l}((\delta+l-1, 1, c, .); 0) \\ &\leq Q_{n-l}((\delta+l-1, b-l+1, b, .); 0). \end{aligned} \tag{B.71}$$

Then, similar to the proof of Case 1, we have

$$Q_{n-l}((\delta+l, 1, c, .); 0) \leq Q_{n-l}((\delta+l, b-l+1, b, .); 0). \tag{B.72}$$

Applying Lemma 13 again, we have

$$\mathbb{E}[V^{\alpha}_{n-l+1}(\delta+l,1,c,\bar{b})] \leq \mathbb{E}[V^{\alpha}_{n-l+1}(\delta+l,b-l+1,b,\bar{b})]. \tag{B.73}$$

We then apply Corollary 14 recursively to obtain

$$\mathbb{E}[V^{\alpha}_{n-1}(\delta+2,l-1,c,\bar{b})] \leq \mathbb{E}[V^{\alpha}_{n-1}(\delta+2,b-1,b,\bar{b})], \tag{B.74}$$

which indicates $\Delta Q^{1,0}_{n-1}(\mathbf{s}') \geq 0$ according to (B.62)-(B.63).

Combining both cases, we thus arrive at the conclusion that the optimal decision at state $\mathbf{s}'$ is to skip as well.

Bibliography

[1] Lee, E. A. (2008) "Cyber Physical Systems: Design Challenges," in *2008 11th IEEE International Symposium on Object and Component-Oriented Real-Time Distributed Computing (ISORC)*, pp. 363–369.

[2] Pasqualetti, F., F. DÃűrfler, and F. Bullo (2013) "Attack Detection and Identification in Cyber-Physical Systems," *IEEE Transactions on Automatic Control*, **58**(11), pp. 2715–2729.

[3] Joo, S., J. Kim, and C. Liu (2007) "Empirical Analysis of the Impact of 2003 Blackout on Security Values of U.S. Utilities and Electrical Equipment Manufacturing Firms," *IEEE Transactions on Power Systems*, **22**(3), pp. 1012–1018.

[4] Tate, J. E. and T. J. Overbye (2008) "Line outage detection using phasor angle measurements," *IEEE Transactions on Power Systems*, **23**(4), pp. 1644–1652.

[5] ——— (2009) "Double line outage detection using phasor angle measurements," in *2009 IEEE Power Energy Society General Meeting*, pp. 1–5.

[6] Abdelaziz, A. Y., S. F. Mekhamer, M. Ezzat, and E. F. El-Saadany (2012) "Line outage detection using support Vector Machine (SVM) based

on the Phasor Measurement Units (PMUs) technology," in *IEEE Power and Energy Society General Meeting*, pp. 1–8.

[7] Emami, R. and A. Abur (2010) "Tracking changes in the external network model," in *North American Power Symposium (NAPS)*, pp. 1–6.

[8] Guler, T., G. Gross, and M. Liu (2007) "Generalized Line Outage Distribution Factors," *IEEE Transactions on Power Systems*, **22**(2), pp. 879–881.

[9] Guo, J., Y. Fu, Z. Li, and M. Shahidehpour (2009) "Direct Calculation of Line Outage Distribution Factors," *IEEE Transactions on Power Systems*, **24**(3), pp. 1633–1634.

[10] Zhu, H. and G. B. Giannakis (2012) "Sparse overcomplete representations for efficient identification of power line outages," *IEEE Transactions on Power Systems*, **27**(4), pp. 2215–2224.

[11] He, M. and J. Zhang (2011) "A Dependency Graph Approach for Fault Detection and Localization Towards Secure Smart Grid," *IEEE Transactions on Smart Grid*, **2**(2), pp. 342–351.

[12] Babakmehr, M., M. G. SimÃŢes, M. B. Wakin, A. A. Durra, and F. Harirchi (2015) "Smart grid topology identification using sparse recovery," in *Industry Applications Society Annual Meeting, 2015 IEEE*, pp. 1–8.

[13] Donmez, B. and A. Abur (2016) "Sparse estimation based external system line outage detection," in *2016 Power Systems Computation Conference (PSCC)*, pp. 1–6.

[14] Feng, G. and A. Abur (2016) "Fault Location Using Wide-Area Measurements and Sparse Estimation," *IEEE Transactions on Power Systems*, **31**(4), pp. 2938–2945.

[15] LLC, M. (2016), "BPA Operations Information," .
URL `https://transmission.bpa.gov/Business/Operations/Outages/default.aspx`

[16] Cevher, V., P. Indyk, C. Hegde, and R. G. Baraniuk (2009) *Recovery of clustered sparse signals from compressive measurements*, *Tech. rep.*, DTIC Document.

[17] Baraniuk, R. G., V. Cevher, M. F. Duarte, and C. Hegde (2010) "Model-Based Compressive Sensing," *IEEE Transactions on Information Theory*, **56**(4), pp. 1982–2001.

[18] Chen, M.-S. and W. E. Dillon (1974) "Power system modeling," *Proceedings of the IEEE*, **62**(7), pp. 901–915.

[19] Needell, D. and J. A. Tropp (2010) "CoSaMP: iterative signal recovery from incomplete and inaccurate samples," *Communications of the ACM*, **53**(12), pp. 93–100.

[20] Hegde, C., P. Indyk, and L. Schmidt (2015) "A nearly-linear time framework for graph-structured sparsity," in *Proceedings of the 32nd International Conference on Machine Learning*, pp. 928–937.

[21] Karp, R. M. (1972) "Reducibility Among Combinatorial Problems," in *Proceedings of a symposium on the Complexity of Computer Computations, March 20-22, 1972, IBM Thomas J. Watson, New York.*, pp. 85–103.

[22] Ljubic, I., R. Weiskircher, U. Pferschy, G. W. Klau, P. Mutzel, and M. Fischetti (2005) "Solving the prize-collecting Steiner tree problem to optimality," in *ALENEX/ANALCO*, pp. 68–76.

[23] Goemans, M. X. and D. P. Williamson (1995) "A general approximation technique for constrained forest problems," *SIAM Journal on Computing*, **24**(2), pp. 296–317.

[24] Tibshirani, R. (1996) "Regression shrinkage and selection via the Lasso," *Journal of the Royal Statistical Society. Series B (Methodological)*, pp. 267–288.

[25] Meinshausen, N. and P. Bühlmann (2006) "High-dimensional graphs and variable selection with the Lasso," *The annals of statistics*, pp. 1436–1462.

[26] Luo, Z.-Q. (2003) "Applications of convex optimization in signal processing and digital communication," *Mathematical programming*, **97**(1), pp. 177–207.

[27] Zimmerman, R. D., C. E. Murillo-Sánchez, and D. Gan (1997) "MATPOWER: A MATLAB power system simulation package," *Manual, Power Systems Engineering Research Center, Ithaca NY*, **1**.

[28] Ceraolo, M. and D. Poli (2014) *Large Power Systems: Structure and Operation*, IEEE.
URL `https://ieeexplore.ieee.org/document/6817743`

[29] Kim, D., T. Y. Chun, S. Yoon, G. Lee, and Y. Shin (2017) "Wavelet-based event detection method using PMU data," in *2017 IEEE Power Energy Society General Meeting*, pp. 1–1.

[30] Cui, M., J. Wang, J. Tan, A. R. Florita, and Y. Zhang (2019) "A Novel Event Detection Method Using PMU Data With High Precision," *IEEE Transactions on Power Systems*, **34**(1), pp. 454–466.

[31] Zhang, J. Y. and C. M. Bush (2016) "PMU based islanding detection to improve system operation," in *2016 IEEE Power and Energy Society General Meeting (PESGM)*, pp. 1–5.

[32] Wen, S., Y. Wang, Y. Tang, Y. Xu, P. Li, and T. Zhao (2019) "Real-Time Identification of Power Fluctuations Based on LSTM Recurrent Neural Network: A Case Study on Singapore Power System," *IEEE Transactions on Industrial Informatics*, **15**(9), pp. 5266–5275.

[33] Ren, H., Z. Hou, and P. Etingov (2018) "Online Anomaly Detection Using Machine Learning and HPC for Power System Synchrophasor Measurements," in *2018 IEEE International Conference on Probabilistic Methods Applied to Power Systems (PMAPS)*, IEEE, pp. 1–5.

[34] Gajjar, G. and S. A. Soman (2014) "Auto detection of power system events using wide area frequency measurements," in *2014 Eighteenth National Power Systems Conference (NPSC)*, pp. 1–6.

[35] Chen, Y. C., T. Banerjee, A. D. Domínguez-García, and V. V. Veeravalli (2016) "Quickest Line Outage Detection and Identification," *IEEE Transactions on Power Systems*, **31**(1), pp. 749–758.

[36] Huang, Y., J. Tang, Y. Cheng, H. Li, K. A. Campbell, and Z. Han (2016) "Real-Time Detection of False Data Injection in Smart Grid Networks: An Adaptive CUSUM Method and Analysis," *IEEE Systems Journal*, **10**(2), pp. 532–543.

[37] Yi Huang, H. Li, K. A. Campbell, and Zhu Han (2011) "Defending false data injection attack on smart grid network using adaptive CUSUM test," in *2011 45th Annual Conference on Information Sciences and Systems*, pp. 1–6.

[38] Samantaray, S. R., A. Samui, and B. Chitti Babu (2010) "S-transform based cumulative sum detector (CUSUM) for islanding detection in Distributed Generations," in *2010 Joint International Conference on Power Electronics, Drives and Energy Systems 2010 Power India*, pp. 1–6.

[39] He, X., M. Pun, C. . J. Kuo, and Y. Zhao (2010) "A Change-Point Detection Approach to Power Quality Monitoring in Smart Grids," in *2010 IEEE International Conference on Communications Workshops*, pp. 1–5.

[40] Malhotra, P., L. Vig, G. Shroff, and P. Agarwal (2015) "Long short term memory networks for anomaly detection in time series," in *Proceedings*, Presses universitaires de Louvain, p. 89.

[41] Hayton, P., S. Utete, D. King, S. King, P. Anuzis, and L. Tarassenko (2006) "Static and dynamic novelty detection methods for jet engine health monitoring," *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, **365**(1851), pp. 493–514.

[42] Wang, S., S. Bi, Y. Jun, Zhang, and J. Huang (2017) "Electrical Vehicle Charging Station Profit Maximization: Admission, Pricing, and Online Scheduling," *ArXiv e-prints*, 1705.02116.

[43] Maigha and M. L. Crow (2017) "Cost-Constrained Dynamic Optimal Electric Vehicle Charging," *IEEE Transactions on Sustainable Energy*, **8**(2), pp. 716–724.

[44] Shiryaev, A. N. (1961) "The problem of the most rapid detection of a disturbance in a stationary process," .

[45] Page, E. S. (1954) "Continuous inspection schemes," *Biometrika*, **41**(1/2), pp. 100–115.

[46] Lorden, G. (1971) "Procedures for Reacting to a Change in Distribution," *Ann. Math. Statist.*, **42**(6), pp. 1897–1908.
URL `https://doi.org/10.1214/aoms/1177693055`

[47] Rao, T. S. and M. Gabr (1980) "A test for linearity of stationary time series," *Journal of time series analysis*, **1**(2), pp. 145–158.

[48] Oklahoma Gas & Electric Company, `https://www.oge.com`.

[49] Chan, C. C. (2002) "The state of the art of electric and hybrid vehicles," *Proceedings of the IEEE*, **90**(2), pp. 247–275.

[50] Sallabi, F., K. Shuaib, and M. Alahmad (2017) "Online scheduling scheme for smart electric vehicle charging infrastructure," in *13th International Wireless Communications and Mobile Computing Conference (IWCMC)*, pp. 1297–1302.

[51] Clement-Nyns, K., E. Haesen, and J. Driesen (2010) "The Impact of Charging Plug-In Hybrid Electric Vehicles on a Residential Distribution Grid," *IEEE Transactions on Power Systems*, **25**(1), pp. 371–380.

[52] Ghavami, A., K. Kar, and A. Gupta (2016) "Decentralized Charging of Plug-in Electric Vehicles With Distribution Feeder Overload Control," *IEEE Transactions on Automatic Control*, **61**(11), pp. 3527–3532.

[53] Gan, L., U. Topcu, and S. H. Low (2013) "Optimal decentralized protocol for electric vehicle charging," *IEEE Transactions on Power Systems*, **28**(2), pp. 940–951.

[54] Ma, Z., D. S. Callaway, and I. A. Hiskens (2013) "Decentralized Charging Control of Large Populations of Plug-in Electric Vehicles," *IEEE Transactions on Control Systems Technology*, **21**(1), pp. 67–78.

[55] Lacey, G., G. Putrus, and E. Bentley (2017) "Smart EV charging schedules: supporting the grid and protecting battery life," *IET Electrical Systems in Transportation*, **7**(1), pp. 84–91.

[56] Song, Y., Y. Zheng, and D. J. Hill (2017) "Optimal Scheduling for EV Charging Stations in Distribution Networks: A Convexified Model," *IEEE Transactions on Power Systems*, **32**(2), pp. 1574–1575.

[57] Korolko, N. and Z. Sahinoglu (2017) "Robust Optimization of EV Charging Schedules in Unregulated Electricity Markets," *IEEE Transactions on Smart Grid*, **8**(1), pp. 149–157.

[58] Chekired, D. A. E., S. Dhaou, L. Khoukhi, and H. T. Mouftah (2017) "Dynamic pricing model for EV charging-discharging service based on cloud computing scheduling," in *2017 13th International Wireless Communications and Mobile Computing Conference (IWCMC)*, pp. 1010–1015.

[59] Chen, Q., N. Liu, C. Wang, and J. Zhang (2014) "Optimal power utilizing strategy for PV-based EV charging stations considering Real-time price," in *2014 IEEE Conference and Expo Transportation Electrification Asia-Pacific (ITEC Asia-Pacific)*, pp. 1–6.

[60] Yang, Z., L. Sun, M. Ke, Z. Shi, and J. Chen (2014) "Optimal Charging Strategy for Plug-In Electric Taxi With Time-Varying Profits," *IEEE Transactions on Smart Grid*, **5**(6), pp. 2787–2797.

[61] Tang, W., S. Bi, and Y. J. . Zhang (2014) "Online Coordinated Charging Decision Algorithm for Electric Vehicles Without Future Information," *IEEE Transactions on Smart Grid*, **5**(6), pp. 2810–2824.

[62] Stein, S., E. Gerding, V. Robu, and N. R. Jennings (2012) "A Model-based Online Mechanism with Pre-commitment and Its Application to Electric Vehicle Charging," in *Proceedings of the 2012 International Conference on Autonomous Agents and Multiagent Systems - Volume 2*, pp. 669–676.

[63] Ströhle, P., E. H. Gerding, M. M. de Weerdt, S. Stein, and V. Robu (2014) "Online Mechanism Design for Scheduling Non-preemptive Jobs Under Uncertain Supply and Demand," in *Proceedings of the 2014 International Conference on Autonomous Agents and Multi-agent Systems*, pp. 437–444.

[64] Zhang, T., W. Chen, Z. Han, and Z. Cao (2014) "Charging Scheduling of Electric Vehicles With Local Renewable Energy Under Uncertain Electric Vehicle Arrival and Grid Power Price," *IEEE Transactions on Vehicular Technology*, **63**(6), pp. 2600–2612.

[65] Qin, H. and W. Zhang (2011) "Charging Scheduling with Minimal Waiting in a Network of Electric Vehicles and Charging Stations," in *Proceedings of*

the Eighth ACM International Workshop on Vehicular Inter-networking, pp. 51–60.

[66] Zhu, M., X. Y. Liu, L. Kong, R. Shen, W. Shu, and M. Y. Wu (2014) "The charging-scheduling problem for electric vehicle networks," in *2014 IEEE Wireless Communications and Networking Conference (WCNC)*, pp. 3178–3183.

[67] Wang, Y. and J. Thompson (2017) "Admission and scheduling mechanism for electric vehicle charging with renewable energy," in *2017 IEEE International Conference on Communications Workshops (ICC Workshops)*, pp. 1304–1309.

[68] Wei, Z., J. He, M. Xing, and L. Cai (2015) "Utility maximization for Electric Vehicle charging with admission control and scheduling," in *2015 IEEE International Conference on Communications (ICC)*, pp. 661–666.

[69] Zhang, L. and Y. Li (2017) "Optimal Management for Parking-Lot Electric Vehicle Charging by Two-Stage Approximate Dynamic Programming," *IEEE Transactions on Smart Grid*, **8**(4), pp. 1722–1730.

[70] Maigha and M. L. Crow (2017) "Cost-Constrained Dynamic Optimal Electric Vehicle Charging," *IEEE Transactions on Sustainable Energy*, **8**(2), pp. 716–724.

[71] Chen, S., L. Tong, and T. He (2011) "Optimal deadline scheduling with commitment," in *49th Annual Allerton Conference on Communication, Control, and Computing (Allerton)*, pp. 111–118.

[72] Yu, Z., S. Chen, and L. Tong (2016) "An intelligent energy management system for large-scale charging of electric vehicles," *CSEE Journal of Power and Energy Systems*, **2**(1), pp. 47–53.

[73] Gushchin, A., S. H. Tseng, and A. Tang (2016) "Optimization-based network flow deadline scheduling," in *IEEE 24th International Conference on Network Protocols (ICNP)*, pp. 1–10.

[74] Karmarkar, N. (1984) "A New Polynomial-time Algorithm for Linear Programming," in *Proceedings of the Sixteenth Annual ACM Symposium on Theory of Computing*, pp. 302–311.

[75] Brucker, P. (2001) *Scheduling Algorithms*, Springer, Secaucus, NJ, USA.

[76] Kaul, S. K., R. D. Yates, and M. Gruteser (2012) "Real-time status: How often should one update?" in *IEEE INFOCOM*, Orlando, FL, USA, pp. 2731–2735.

[77] ——— (2012) "Status updates through queues," in *Conference on Information Sciences and Systems (CISS)*, Princeton, NJ, USA, pp. 1–6.

[78] Yates, R. D. and S. K. Kaul (2012) "Real-time status updating: Multiple sources," in *IEEE International Symposium on Information Theory (ISIT)*, Cambridge, MA, USA, pp. 2666–2670.

[79] ——— (2016) "The Age of Information: Real-Time Status Updating by Multiple Sources," *ArXiv e-prints*.
URL `http://arxiv.org/abs/1608.08622`

[80] Pappas, N., J. Gunnarsson, L. Kratz, M. Kountouris, and V. Angelakis (2015) "Age of information of multiple sources with queue management," in *IEEE International Conference on Communications (ICC)*, pp. 5935–5940.

[81] Najm, E. and R. Nasser (2016) "Age of information: The gamma awakening," in *IEEE International Symposium on Information Theory (ISIT)*, Barcelona, Spain, pp. 2574–2578.

[82] Kam, C., S. Kompella, G. D. Nguyen, J. E. Wieselthier, and A. Ephremides (2016) "Age of information with a packet deadline," in *IEEE International Symposium on Information Theory (ISIT)*, Barcelona, Spain, pp. 2564–2568.

[83] Chen, K. and L. Huang (2016) "Age-of-information in the presence of error," in *IEEE International Symposium on Information Theory (ISIT)*, Barcelona, Spain, pp. 2579–2583.

[84] Kaul, S. K. and R. D. Yates (2018) "Age of Information: Updates with Priority," in *IEEE International Symposium on Information Theory (ISIT)*, pp. 2644–2648.

[85] Kam, C., S. Kompella, and A. Ephremides (2013) "Age of information under random updates," in *IEEE International Symposium on Information Theory (ISIT)*, Istanbul, Turkey, pp. 66–70.

[86] ——— (2014) "Effect of message transmission diversity on status age," in *IEEE International Symposium on Information Theory (ISIT)*, Honolulu, HI, USA, pp. 2411–2415.

[87] Kam, C., S. Kompella, G. D. Nguyen, and A. Ephremides (2016) "Effect of Message Transmission Path Diversity on Status Age," **62**(3), pp. 1360–1374.

[88] Yates, R. D. (2018) "Status Updates through Networks of Parallel Servers," in *IEEE International Symposium on Information Theory (ISIT)*, pp. 2281–2285.

[89] Zhong, J., R. D. Yates, and E. Soljanin (2018) "Multicast with Prioritized Delivery: How Fresh is Your Data?" in *IEEE 19th International Workshop on Signal Processing Advances in Wireless Communications (SPAWC)*, pp. 1–5.

[90] Costa, M., M. Codreanu, and A. Ephremides (2014) "Age of information with packet management," in *IEEE International Symposium on Information Theory (ISIT)*, Honolulu, HI, USA, pp. 1583–1587.

[91] ——— (2016) "On the Age of Information in Status Update Systems With Packet Management," **62**(4), pp. 1897–1910.

[92] Huang, L. and E. Modiano (2015) "Optimizing age-of-information in a multi-class queueing system," in *IEEE International Symposium on Information Theory (ISIT)*, Hong Kong, China, pp. 1681–1685.

[93] Sun, Y., E. Uysal-Biyikoglu, R. D. Yates, C. E. Koksal, and N. B. Shroff (2016) "Update or wait: How to keep your data fresh," in *IEEE INFOCOM*, San Francisco, CA, USA, pp. 1–9.

[94] Sun, Y., Y. Polyanskiy, and E. Uysal-Biyikoglu (2017) "Remote Estimation of the Wiener Process over a Channel with Random Delay," *CoRR*, **abs/1701.06734**.

[95] Yates, R. D. (2015) "Lazy is timely: Status updates by an energy harvesting source," in *IEEE International Symposium on Information Theory (ISIT)*, Hong Kong, China, pp. 3008–3012.

[96] Bacinoglu, B. T., E. T. Ceran, and E. Uysal-Biyikoglu (2015) "Age of information under energy replenishment constraints," in *Information Theory and Applications Workshop*, San Diego, CA, USA, pp. 25–31.

[97] Wu, X., J. Yang, and J. Wu (2018) "Optimal Status Update for Age of Information Minimization with an Energy Harvesting Source," **2**(1), pp. 193 – 204.

[98] Bacinoglu, B. T. and E. Uysal-Biyikoglu (2017) "Scheduling Status Updates to Minimize Age of Information with an Energy Harvesting Sensor," in *IEEE International Symposium on Information Theory (ISIT)*.

[99] Arafa, A., J. Yang, and S. Ulukus (2018) "Age-minimal online policies for energy harvesting sensors with random battery recharges," in *IEEE International Conference on Communications (ICC)*.

[100] Arafa, A., J. Yang, S. Ulukus, and V. Poor (2018) "Age-Minimal Online Policies for Energy Harvesting Sensors with Incremental Battery Recharges," in *Information Theory and Applications Workshop*, San Diego, CA, USA.

[101] Tan Bacinoglu, B., Y. Sun, E. Uysal-Biyikoglu, and V. Mutlu (2018) "Achieving the Age-Energy Tradeoff with a Finite-Battery Energy Harvesting Source," in *IEEE International Symposium on Information Theory (ISIT)*.

[102] Feng, S. and J. Yang (2018) "Optimal Status Updating for an Energy Harvesting Sensor with a Noisy Channel," in *IEEE INFOCOM - Workshop on Age of Information*.

[103] ——— (2018) "Minimizing Age of Information for an Energy Harvesting Source with Updating Failures," in *IEEE International Symposium on Information Theory (ISIT)*.

[104] Kadota, I., A. Sinha, E. Uysal-Biyikoglu, R. Singh, and E. Modiano (2018) "Scheduling Policies for Minimizing Age of Information in Broadcast Wireless Networks," *ArXiv e-prints*, 1801.01803.

[105] Talak, R., S. Karaman, and E. Modiano (2018) "Optimizing Information Freshness in Wireless Networks under General Interference Constraints," in *International Symposium on Mobile Ad Hoc Networking and Computing(MobiHoc)*.

[106] Hsu, Y. P., E. Modiano, and L. Duan (2017) "Age of information: Design and analysis of optimal scheduling algorithms," in *IEEE International Symposium on Information Theory (ISIT)*, pp. 561–565.

[107] Hsu, Y.-P. (2018) "Age of Information: Whittle Index for Scheduling Stochastic Arrivals," *ArXiv e-prints*, 1801.03422.

[108] Kadota, I., A. Sinha, and E. Modiano (2018) "Optimizing Age of Information in Wireless Networks with Throughput Constraints," in *IEEE INFOCOM*.

[109] He, Q., D. Yuan, and A. Ephremides (2018) "Optimal Link Scheduling for Age Minimization in Wireless Systems," *IEEE Transactions on Information Theory*, **64**(7), pp. 5381–5394.

[110] Najm, E. and R. Nasser (2016) "Age of information: The gamma awakening," in *IEEE International Symposium on Information Theory (ISIT)*, pp. 2574–2578.

[111] Bedewy, A. M., Y. Sun, and N. B. Shroff (2019) "Minimizing the Age of Information through Queues," *IEEE Transactions on Information Theory*, pp. 1–1.

[112] ——— (2017) "Age-optimal information updates in multihop networks," in *IEEE International Symposium on Information Theory (ISIT)*, pp. 576–580.

[113] Sun, Y., E. Uysal-Biyikoglu, and S. Kompella (2018) "Age-optimal updates of multiple information flows," in *IEEE INFOCOM - Workshop on Age of Information*, pp. 136–141.

[114] Najm, E., R. Yates, and E. Soljanin (2017) "Status updates through M/G/1/1 queues with HARQ," in *IEEE International Symposium on Information Theory (ISIT)*, pp. 131–135.

[115] Farazi, S., A. G. Klein, and D. R. Brown (2018) "Age of Information in Energy Harvesting Status Update Systems: When to Preempt in Service?"

in *2018 IEEE International Symposium on Information Theory (ISIT)*, pp. 2436–2440.

[116] Kavitha, V., E. Altman, and I. Saha (2018) "Controlling Packet Drops to Improve Freshness of information," *ArXiv e-prints*, 1807.09325.

[117] Bertsekas, D. P. (2000) *Dynamic Programming and Optimal Control*, 2nd ed., Athena Scientific.

[118] Sennott, L. I. (1997) "On computing average cost optimal policies with application to routing to parallel queues," *Mathematical Methods of Operations Research*, **45**(1), pp. 45–62.
URL https://doi.org/10.1007/BF01194247

[119] Puterman, M. L. (2014) *Markov decision processes: discrete stochastic dynamic programming*, John Wiley & Sons.

[120] Kellerer, H., U. Pferschy, and D. Pisinger (2004) *Knapsack Problems*, Springer Berlin Heidelberg, Berlin, Heidelberg, pp. 73–93.

Vita

Boyu Wang

Boyu Wang received his B.S. and M.S. degrees in Electrical Engineering from Beijing Jiaotong University, Beijing, China in 2011 and 2014, respectively. He received another M.S. degree in Electrical Engineering from University of Arkansas, Fayetteville, in 2016. He has been a research assistant at Wireless Communications and Networking (WCAN) Lab in the Pennsylvania State University from 2016 to 2019, where he obtains his Ph.D. degree in December 2019. His research interests include machine learning, scheduling, optimization and signal processing in networked systems.

Selected Publications:

1. Boyu Wang, Chao Gan, Jing Yang, Chinmay Hedge and Jingxian Wu *"**Graph-based Multiple-Line Outage Identification in Power Transmission Systems**"*, in IEEE Power Energy Society General Meeting (PES), Chicago, IL, July 2017.

2. Boyu Wang, Jing Yang *"**Optimal Electric Vehicle Charging Scheduling with Time-varying Profits**"*, in 52th Annual Conference on Information Sciences and Systems (CISS), Princeton, NJ, USA, March 2018.

3. Boyu Wang, Songtao Feng and Jing Yang *"**To Skip or to Switch? Minimizing Age of Information under Link Capacity Constraint**"*, in 19th IEEE International Workshop on Signal Processing Advances in Wireless Communications (SPAWC 2018), Kalamata, Greece, June 2018.

4. Boyu Wang, Songtao Feng and Jing Yang *"**When to Preempt? Age of Information Minimization under Link Capacity Constraint**"*, in Journal of Communication, Volume:21, Issue:3, 2019.

5. Boyu Wang, Jing Yang *"**Optimal Electric Vehicle Charging Scheduling with Time-varying Profits**"*, IEEE Transaction on Smart Grid. Submitted.

6. Boyu Wang, Chao Gan, Jing Yang, Chinmay Hedge and Jingxian Wu *"**Graph-based Multiple-Line Outage Identification in Power Transmission Systems**"*, IEEE Transaction on Power System. submitted.

7. Boyu Wang, Jing Yang *"**LSTM-based Quick Event Detection in Power Systems** "*, IIEEE Power Energy Society General Meeting (PES). submitted.