# Resource Allocation in Smart Infrastructure: Case Studies in Video Delivery and Electric Power Networks

Submitted in partial fulfillment of the requirements for

the degree of

Doctor of Philosophy

in

Electrical and Computer Engineering

Xiaoqi Yin

B.E., Automation, Tsinghua University

M.S., Electrical and Computer Engineering, Carnegie Mellon University

Carnegie Mellon University
Pittsburgh, PA

August, 2016

Resource Allocation in Smart Infrastructure:

Case Studies in Video Delivery and Electric Power Networks

## ACKNOWLEDGMENTS

First, I would like to extend sincere gratitude to my Ph.D. advisor, Prof. Bruno Sinopoli, who has always been supportive, patient, open-minded, visonary, and inspiring. I could not imagine to have a nicer advisor than Prof. Sinopoli.

Next, special thanks go to two professors who have also significantly shaped my PhD journey, Prof. Marija Ilić and Prof. Vyas Sekar. I enjoyed working with them and also learnt a lot from them. I could have not gone so far without their help.

I want to thank my external committee members, Prof. Adam Wierman and Prof. Magnus Egerstedt, who have always been supportive and provided valuable philosophical, inspirational, and technical feedback to the dissertation.

This dissertation is funded in part by National Science Foundation. I greatly appreciate the financial support from the sponsor.

I also want to thank other faculty who have helped me along this journey, Prof. Rohit Negi, Prof. Radu Marculescu, Prof. Anthony Rowe, Prof. Xin Li, Prof. John Dolan, Prof. José Moura, Prof. Soummya Kar, and Prof. Paul Weng.

Special thanks go to my departmental Ph.D. academic advisors, Ms. Samantha Goldstein and Mr. Nathan Snizaski, for their support and guidance, and to Ms. Claire E. Bauerle and Ms. Carol Patterson, for making Porter Hall B-level a nice workspace.

I have been most fortunate to meet great friends and colleagues on this journey. I wish to thank Yilin Mo, for his sincere friendship and mentorship, Yang Weng, Qixing Liu, Dinghuan Zhu, Luca Parolini, Sabina Zejnilovic, Dragana Bajovic, Rohan Chabukswar, Akshay Rajhans, JY Joo, Sergio Pequito, Yue Ding, Sean Weerakkody, Steven Aday, Xiaofei Liu, Niranjini Rajagopal, Mihovil Bartulovic, John Costanzo, Lucas Balthazar, Xiao Zhang, Liangyan Gui, Xia Miao, Junyao Guo, Yaoqing Yang, Joya Deli, June Zhang, Andrew Hsu, Kyri Baker, Subhro Das, Nipon Popli, Stefanos Baros, Jon Donadee, Javad Mohammadi, Junchen Jiang, and others too numerous to name.

Finally, I want to thank my parents, Xiaochun Su and Haibin Yin, and my girlfriend Rui Yang, for their love and support. They are always standing by my side and are my source of strength on this journey.

# ABSTRACT

Resource allocation schemes play an important role in large-scale smart infrastructures to ensure efficiency and fairness among users. However, designing good resource allocation schemes is challenging due to technical limitation, policy barriers, and cost of change. The goal of this thesis is to develop a methodology to design model-based, principled and practical resource allocation schemes. Given the diverse characteristics of infrastructures, it is difficult to design unified models and algorithms. Instead, we employ a case-study-based approach on two representative smart infrastructures: Internet video delivery and electric power networks. We further generalize the insights and develop a principled qualitative guideline to design resource allocation schemes in smart infrastructures.

The Internet video delivery system employs a protocol-based resource allocation scheme: network bandwidth is implicitly allocated by transport layer protocol (TCP) while client-side video players adapt video quality based on application layer protocol (MPEG-DASH) to optimize users quality of experience (QoE). We study 1) how client-side video players improve users QoE by employing Model Predictive Control-based bitrate adaptation algorithms and 2) how to achieve multiplayer QoE fairness by router-side bandwidth allocation policies. We prototype and evaluate the algorithms in real video players.

On the other hand, market-based schemes are adopted in real-time economic dispatch in electric power systems to satisfy demand by lowest-cost generation. However, such schemes can lead to power imbalances and market inefficiency when slow generators fail to follow system operators command. We study 1) how system operators can mitigate power imbalance by employing a centralized, two-stage robust dispatch and 2) how the market design can be improved by penalizing non-complying generators.

Based on the lessons from the case studies, we develop a general methodology to design resource allocation schemes: First, develop a formal model capturing system objectives, dynamics, and constraints; Second, identify key practical constraints that have major impact on the choice of schemes; Finally, design model-based schemes that respect practical constraints for short-term and obtain insights to inform protocol or market improvement in the long run. We envision that a mathematical theory can significantly improve the future resource allocation ecosystems.

# Contents

**List of Tables** **ix**

**List of Figures** **xi**

**List of Acronyms** **xii**

**1 Introduction** **1**

1.1 Motivation . . . . . . . . . . 1

1.2 Proposed Case-Study-Based Approach . . . . . . . . . . 3

1.3 General Methodologies to Design Resource Allocation Schemes . . . . . . . . 4

1.4 Summary of Contributions . . . . . . . . . . 6

1.5 Thesis Outline . . . . . . . . . . 8

**2 Case Study in Video Delivery Networks** **9**

2.1 Overview . . . . . . . . . . 9

2.2 Background . . . . . . . . . . 13

2.2.1 Single-player Video Streaming . . . . . . . . . . 13

2.2.2 Multi-player Video Streaming . . . . . . . . . . 16

2.3 Single-Player Adaptive Video Streaming . . . . . . . . . . 19

2.3.1 Control-Theoretic Model . . . . . . . . . . 19

2.3.2 Model Predictive Control Approach
for Optimal Bitrate Adaptation . . . . . . . . . . . . . . . . . . . . . 26
2.3.3 Using MPC in Practice — FastMPC . . . . . . . . . . . . . . . . . . 31
2.3.4 Implementation . . . . . . . . . . . . . . . . . . . . . . . . . . . . 33
2.3.5 Evaluation . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 36
2.4 Multi-Player Adaptive Video Streaming . . . . . . . . . . . . . . . . . . 48
2.4.1 Modeling . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 48
2.4.2 Analysis of Fairness in Steady State . . . . . . . . . . . . . . . . . 53
2.4.3 NMPC-based Router-Assisted Bandwidth Allocation for QoE Fairness 60
2.4.4 Evaluation . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 61
2.5 Summary . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 67

**3 Case Study in Electric Power Networks 69**
3.1 Overview . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 69
3.2 Background . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 71
3.2.1 Risks in Today's Electricity Markets . . . . . . . . . . . . . . . . . 72
3.2.2 Risks Posed by Slow Generators' Deviations . . . . . . . . . . . . . 73
3.2.3 Related Work in Electricity Market Formulations . . . . . . . . . . 75
3.3 Centralized Risk-Based Real-Time Dispatch . . . . . . . . . . . . . . . . 76
3.4 Risk-Based Market Design . . . . . . . . . . . . . . . . . . . . . . . . . 79
3.4.1 Market Structure . . . . . . . . . . . . . . . . . . . . . . . . . . . 80
3.4.2 Profit Maximization of Generators under Proposed Market Structure 82
3.4.3 Comparison with Centralized Approach . . . . . . . . . . . . . . . . 83
3.5 Evaluation . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 85
3.5.1 Setup . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 85
3.5.2 System-Wide Impact . . . . . . . . . . . . . . . . . . . . . . . . . 86

3.5.3 Impact on Individual Generators . . . . . . . . . . . . . . . . . . . . . 89

3.5.4 Sensitivity Analysis . . . . . . . . . . . . . . . . . . . . . . . . . . . 89

3.5.5 Summary of Results . . . . . . . . . . . . . . . . . . . . . . . . . . . 92

**4 Conclusions and Future Work** **94**

4.1 Concluding Remarks . . . . . . . . . . . . . . . . . . . . . . . . . . . . 94

4.2 Future Work . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 95

4.3 Toward a Mathematical Theory for Better Resource Allocation Ecosystems . 98

**Bibliography** **101**

# List of Tables

2.1 FastMPC table size . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 46

3.1 Generator profiles: 1-3 are slow generators, 4,5 are fast generators . . . . . . 73

3.2 Risk-based settlement process: $\lambda$ is the RTD price while $\alpha$ is the redispatch price . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 80

# List of Figures

2.1 Abstract model of DASH players . . . . . . . . . . . . . . . . . . . . . . . . 14

2.2 The Internet video delivery ecosystem . . . . . . . . . . . . . . . . . . . . . . 17

2.3 Illustration of buffer dynamics . . . . . . . . . . . . . . . . . . . . . . . . . . 21

2.4 Formulation for QoE maximization ($QOE\_MAX_1^K$) subject to buffer and throughput dynamics. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 23

2.5 Design space of algorithms for the video adaptation problem: Most current approaches choose the bitrate as a function of only one variable; e.g., A1 is rate-based (RB) while A2 is buffer-based (BB). . . . . . . . . . . . . . . . . 25

2.6 "FastMPC" idea: We enumerate possible scenarios and create a table indexing the optimal decision for each scenario. . . . . . . . . . . . . . . . . . . . . . 32

2.7 `dash.js` code structure and our modifications . . . . . . . . . . . . . . . . . 35

2.8 Characteristics of datasets . . . . . . . . . . . . . . . . . . . . . . . . . . . . 38

2.9 Real experiment results with different throughput traces . . . . . . . . . . . 39

2.10 Detailed performance for FCC dataset . . . . . . . . . . . . . . . . . . . . . 41

2.11 Detailed performance for HSDPA dataset . . . . . . . . . . . . . . . . . . . . 41

2.12 Sensitivity analysis vs. operating conditions . . . . . . . . . . . . . . . . . . 44

2.13 MPC configuration parameters . . . . . . . . . . . . . . . . . . . . . . . . . 47

2.14 Modeling multiplayer joint bandwidth allocation and bitrate adaptation problem 48

2.15 Social welfare vs fairness tradeoff . . . . . . . . . . . . . . . . . . . . . . . . 65

2.16 Impact of QoE functions . . . . . . . . . . . . . . . . . . . . . . . . . . . 66
2.17 Impact of initial conditions . . . . . . . . . . . . . . . . . . . . . . . . . . . 67

3.1 Today's online power balancing scheme . . . . . . . . . . . . . . . . . . . . 73
3.2 Supply-demand imbalance as a result of individual generator's deviation from schedule . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 75
3.3 The new risk-based online power balancing scheme . . . . . . . . . . . . . . 76
3.4 The IEEE 24-bus reliability test system (IEEE One Area RTS-96) . . . . . . 85
3.5 System-wide impact of deviations . . . . . . . . . . . . . . . . . . . . . . . . 87
3.6 Example bid function for slow generators . . . . . . . . . . . . . . . . . . . . 90
3.7 Impact of risk preference . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 91
3.8 Impact of redispatch price $\alpha$ . . . . . . . . . . . . . . . . . . . . . . . . . . 92

# List of Acronyms

|  |  |
|---|---|
| **ABR** | Bitrate Adaptation |
| **MPC** | Model Predictive Control |
| **RB** | Rate-Based Algorithms |
| **BB** | Buffer-Based Algorithms |
| **RTD** | Real-Time Dispatch |
| **AGC** | Automatic Generation Control |
| **(I)SO** | (Independent) System Operator |

# Chapter 1

# Introduction

## 1.1 Motivation

Resource allocation schemes play an important role in large-scale smart infrastructures to ensure efficiency and fairness among users. To name a few examples, in electric power networks, least-cost generation resources are scheduled and controlled in real-time to ensure satisfaction of users' demand. In video delivery networks, network bandwidth resources are shared among internet applications to deliver streaming video from the source server to end users. In transportation networks, carefully scheduling and routing are carried out so that passengers share the road network in a congestion-free manner. As such, the efficiency and fairness will suffer without designing good resource allocation schemes.

However, designing optimal resource allocation schemes is difficult due to three key practical challenges: First, *technical limitation* can negatively impact the sensing and control capabilities of the system. For example, without SCADA system, power system operators are not able to measure the system state in real time, which makes real-time control difficult. On the other hand, software-defined network (SDN) has enabled centralized network control that used to be difficult in legacy networks. Second, it may not be available to obtain key state information of the system and the users due to *policy barriers*, such as privacy concerns. For example, while appliance-level energy usage pattern data can significantly improve demand reponse program, obtaining such data may intrude users' private information. Finally, the *cost of change* from legacy sub-optimal resource allocation schemes may be too high to adopt improved schemes. For instance, it is difficult to change widely adopted protocols, such as TCP.

As such, this thesis focus on the following question:

*How to design close-to-optimal and practical resource allocation schemes in smart infrastructures that better utilizes the increased sensing and actuation capabilities while respecting the practical challenges posed by technical overhead, policy barriers and cost of change?*

There are two key questions with respect to the design of such resource allocation schemes:

- Who makes the decision, e.g., by a centralized controller with full actuation capability, or individual entities with limited view of the system?
- How system and user information is exchanged, e.g., how does the system knows the objective, states and dynamics of the users, is the information exchange explicit or implicit?

Depending on the answers to these questions, resource allocation schemes can be roughly divided into three categories: *fully centralized*, *fully distributed*, and *decentralized* schemes. On one hand, a fully centralized resource allocation scheme is optimal to achieve efficiency and fairness given full sensing and actuation capabitities of the system, but can suffer from significant overhead and delay; On the other extreme, a fully distributed scheme is light-weight and can significantly reduce overhead and avoid single point of failure, but can reach sub-optimal allocation due to limited view of the system. It is critical to understand the tradeoff here and design resource allocation schemes that is close-to-optimal and respect practical constraints.

## 1.2 Proposed Case-Study-Based Approach

Our goal in this dissertation is to develop a general methodology to design resource allocation schemes. However, given the fact that each smart infrastructure has distinct structure and characteristic, it is difficult to develop model and algorithm that can apply to all infrastructures. As such, in this dissertation, we propose to tackle the problem using a *case-study-based* approach. Based on the structure of resource allocation schemes, we divide current resource allocation schemes into two main categories: *protocol-based* schemes and *market-based* schemes. We pick representative systems in each of the category, namely, Internet video delivery systems and electric power networks, and further investigate the design, implementation and evaluation of resource allocation schemes that respects the practical constraints in these systems. Based on the insights from these representative case studies, we conclude the thesis by summarizing our findings and developing a general principled methodology on the design of resource allocation schemes.

We first carry out case study in a representative protocol-based system. The Internet video delivery system employs a protocol-based resource allocation scheme, where the Dynamic Adaptive Streaming over HTTP (DASH) protocol is used to allocate network bandwidth resources. With the DASH standard, each video is divided into chunks and encoded with different bitrate levels which allows a client-side player to adapt the bitrate according to network conditions so as to improve users' quality-of-experience (QoE). In this thesis, we first present a control-theoretic approach to design the client-side bitrate adaptation algorithms. We then study the efficiency vs. fairness tradeoff when multiple players compete for bandwidth in a network and develops coordination schemes to achieve fairness.

Different from video delivery networks, the smart grid employs a market-based approach to allocate generation resources in the economic dispatch stage. In real-time dispatch stage where generators are scheduled every 5-10 minutes, the slow generator's deviation from the system operator's command has recently been identified as a new type of uncertainty which may lead to increased stress on automatic generation control (AGC) as well as market inefficiency. In this thesis, we first present a centralized risk-aware economic dispatch scheme that can be employed by a system operator to mitigate the deviation. We then develop a distributed risk-based market structure to manage the risk by the profit maximization of individual generators and compare with the centralized solution.

Note that as practical constraints play an important role in the design of resource allocation schemes, in this thesis we want to design schemes with formal modeling and theoretical foundations while respecting practical constraints. The key contribution of this disseration lies in the modeling and the use of principled approaches to design practical, ready-to-ship resource allocation schemes.

## 1.3 General Methodologies to Design Resource Allocation Schemes

Based on the case studies, we develop general principles and methodologies to design resource allocation schemes in smart infrastucture. The key steps are:

1. *Developing a formal mathematical model that captures system objectives, dynamics, and constraints.* Such models can be employed to 1) obtain insights on the structure of the problem, design space, and optimal solution, 2) unify existing practical solutions under the same umbrella and analyze their limitations, and 3) serve as a foundation to adopt state-of-art algorithms from control and optimization theory to solve the problem. The mathematical model can be based on physics, if the system behaviour is well understood, or data-driven, if otherwise. Note that the model can also be hybrid, e.g., in client-side video bitrate adaptation, the video downloading and rendering model

is well understood and based on physics, while the user QoE function and bandwidth dynamics models are data driven.

2. *Identify key practical constraints that have major impact on the choice of resource allocation schemes.* Once we have a model and better understand the design space, it is critical to systematically identify key practical constraints, such as technical overhead, policy barriers and legacy cost, as the practical constraints may significantly limit the available choices of resource allocation schemes.

3. *Design model-based resource allocation schemes that respect practical constraints for short-term solution.* Given the formal model and practical constraints, we can develop model-based, principled but practical solutions for resource allocation that is close-to-optimal while satisfying the practical constraints. Note that such solutions can be off-the-shelf solutions from control and optimization theory based on the model, or approximate solutions to the optimal solutions if optimality needs to be compromised to satisfy practical constraints.

4. *Obtain insights that inform better long-term protocol or market change.* While short-term solutions can be developed that respects practical constraints, it will benefit the system if the practical constraints can be relaxed in the long term. Based on the modeling and analysis, we can quantify the impact of relaxing the practical constraints, and use it to drive protocol or market change in the long run.

To illustrate how these principles work throughout this thesis and in general practical resource allocation problems, let us consider the design of single-player, bitrate adaptation schemes in video delivery networks as an example. We start by building a mathematical model of the bitrate adaptation problem by formulating the objective — quality-of-experience (QoE), video buffer dynamics, and player constraints. While we have physical model of buffer dynamics, the QoE functions do depend on user behaviour and are data-driven. This formal mathematical model allows us to view the existing rate-based and buffer-based algorithms in a unified framework and see their key limitations in that they discard possibly useful information. Based on the insights, we employ off-the-shelf model predictive control (MPC) [74, 75] approach to tackle the problem. However, given the practical constraint that the player code needs to be downloaded and embedded in the browser and the licensing issue, we developed an approximate but fast and practical solution FastMPC [74, 75] to solve the problem with very low overhead. At last, based on the modeling and analysis, we obtain insights for protocol design: The performance of MPC will be significantly improved if the chunk sizes are given in the manifest file in MPEG-DASH standard and a better throughput prediction can be provided by a global throughput predictor.

In summary, the results of our case-study-based approach are model-based, principled but practical solutions to resource allocation schemes in video delivery and electric power networks, as well as general methodologies to develop such schemes that applies to general resource allocation problems. As such, we hope this thesis not only addresses resource allocation problems in specific infrastructures, but also provides useful insights to inform design of resource allocation schemes in general smart infrastructure.

## 1.4 Summary of Contributions

The main contributions of this thesis are summarized as follows:

**Video delivery networks—Single-player adaptive video streaming:** The main contributions are:

- Development of a formal control-theoretic model of the client-side bitrate adaptation problem that unifies prior work under the same umbrella.

- Design and evaluation of a robust single-player client-side bitrate adaptation algorithm based on model predictive control (MPC) that subsumes existing rate- and buffer-based strategies.
- A practical and fast table enumeration based algorithm FastMPC that near-optimally approximates the performance of an exact MPC approach and a low-overhead implementation based on the open source reference video player `dash.js`;

**Video delivery networks—Multi-player adaptive video streaming:** The main contributions are:

- Theoretical analysis of the multi-player QoE fairness of existing bandwidth allocation and bitrate adaptation schemes.
- Modeling, design and evaluation of a router-assisted bandwidth allocation approach that considers client-side bitrate adaptation strategies.

**Electric power networks:** The main contributions are:

- Identification and impact analysis of the non-complying slow generators in the stage of real-time economic dispatch in electric power systems.
- Development of a centralized, two-stage risk-based real-time dispatch scheme that can be employed by system operators to mitigate the negative impact of slow generators' deviations.
- Design and evaluation of a risk-based market clearing scheme that allows fair cost allocation.

**Publications:** Part of the results presented in this disseration have been published as listed below. We publish the results of MPC-based single-player bitrate adaptation algorithms in

ACM HotNets 2014 [75] and ACM SIGCOMM 2015 [74]. While not included in this thesis, an extension of this work is published in ACM SIGCOMM 2016 [68] on design and evaluation of network throughput prediction algorithms. We publish the results on centralized, risk-aware real-time dispatch in the session of Best Conference Papers on Integrated Power System Operations in IEEE PES General Meeting 2014 [44] and results on risk-based market design in ISGT 2015 [73].

## 1.5 Thesis Outline

The rest of this thesis is organized as follows: We first conduct case study in video delivery networks in Chapter 2 and investigate the design and evaluation of single-player and multiplayer adaptive video streaming problem. Next, we move on to case study in electric power networks in Chapter 3 where we focus on designing resource allocation schemes to mitigate the negative impact of slow generators' deviations in real-time economic dispatch stage. Finally, we conclude and discuss future work in Chaper 4.

# Chapter 2

# Case Study in Video Delivery Networks

## 2.1 Overview

Many recent studies have highlighted the critical role that user-perceived quality-of-experience (QoE) plays in Internet video applications, as it ultimately affects revenue streams for content providers [33, 50]. Specifically, metrics such as the duration of rebuffering (i.e., the player's playout buffer does not have content to render), startup delay (i.e., the lag between the user clicking vs. the time to begin rendering), the average playback bitrate, and the variability of the bitrate delivered have emerged as key factors.

Given the complex Internet video delivery ecosystem and presence of diverse bottlenecks, the *bitrate adaptation logic* in the client-side video player becomes critical to optimize user experience [18]. In the HTTP-based delivery model that predominates today [64], videos are typically chunked and encoded at different bitrate levels. The goal of an adaptive video player is to choose the bitrate level for future chunks to deliver the highest possible QoE;

e.g., maximizing bitrate while minimizing the likelihood of rebuffering and avoiding too many bitrate switches.

Many recent efforts have pointed out key challenges in designing this adaptation logic (e.g., [20, 42, 47, 67]) and several proposals have emerged to try and address these challenges (e.g., [20, 43, 47]). Despite the proliferation of numerous algorithms, however, there appears to be a lack of clarity and consensus across these solutions on several fronts; e.g., some argue for better throughput estimation [69], while others suggest improving chunk scheduling [47]. Some researchers even argue against *rate-based* approaches that rely on throughput estimates from previous chunk downloads and make the case for *buffer-occupancy based* algorithms that make their decisions purely based on buffer occupancy [43].

In order to understand the fundamental tradeoffs between different classes of algorithms (e.g., rate- vs buffer-based) under different operating regimes (e.g., low vs. high throughput variability), we begin by formulating the video bitrate adapdation as a *stochastic optimal control* problem. We formally define the key dynamic variables involved in the video adaptation problem and a concrete objective. This framework allows us to outline the broader design space of control algorithms for this problem. We identify a key shortcoming in existing approaches that rely exclusively on pure rate- or buffer-based strategies, and that might be potentially missing out on strategies that *combine* both signals.

Building on insights from the control-theoretic formulation, we argue that *model predictive control (MPC)* [28] is a suitable class of algorithms that can optimally combine both rate-based and buffer-based feedback signals. At a high level, MPC attempts to predict key environment variables over a moving look-ahead horizon and solve an exact optimization problem based on the prediction. MPC is the technology of choice in a multitude of real world control problems [28]. In addition to its intuitive formulation, it can explicitly handle complex control objectives and constraints, and has a set of well understood tuning parameters such as the prediction horizon. Moreover, MPC has other qualitative advantages as its development time is much shorter compared to advanced control methods and it is easier to maintain, as changing model parameters does not require complete redesign.

In our context, the MPC approach entails predicting the expected throughput for the next few chunks and using this to make optimal bitrate decisions for QoE maximization. Indeed, our simulation results confirm that if we could run an optimal MPC algorithm and the prediction error was low, then the MPC scheme can outperform traditional rate-based and buffer-based strategies.

In practice, however, running a MPC-based algorithm is challenging because it needs to solve a non-trivial discrete optimization problem at each time step. Even ignoring the computational overhead, there are practical difficulties as we might need to bundle this solver logic with every video player or require users to download and install additional software. To address these challenges, we develop a simple-yet-efficient *FastMPC* mechanism. Conceptually, FastMPC essentially follows a table enumeration approach, where we describe the problem state-space, solve the specific instances optimally offline, and store the optimal control decisions for future online use. If implemented naively, however, the size of this table can induce significant memory overhead and startup delays for video players (e.g., additional JavaScript to load). Fortunately, we show that with a simple value binning and compression strategy, we can achieve near-optimal performance with manageable table sizes.

We have prototyped our FastMPC bitrate adaptation algorithm in an open source dynamic adaptive streaming player called `dash.js` [1]. Our choice of platform is a pragmatic one—it is the reference open-source implementation for the MPEG-DASH standard based on the HTML5 specification and is actively supported by leading industry participants [7]. We show that our implementation adds negligible overhead to the baseline `dash.js` player. We also showcase the FastMPC-based player in our demo page [16].

We evaluate our algorithms and prototype implementation using realistic emulation experiments on measured [9, 11] and synthetic throughput variability traces. We also augment these results with simulation-based sensitivity analysis experiments to analyze the effect of key operating parameters on the performance of different classes of algorithms.

While client-side bitrate adaptation is critical to ensure high QoE for single player regarding available bandwidth as given by a black box, as video traffic becomes predominant on the internet, it is more and more likely that multiple video players will share bottlenecks and compete for bandwidth in the network [19, 42]. Such scenarios can be seen in home network, commercial building network, and campus networks, where multiple devices (e.g., HDTV, tablet, laptop, cell phone, etc.) connect to Internet by a single Wifi router. In these cases, in addition to single-player QoE, the multi-player QoE fairness becomes a critical issue.

While there have been several practical proposals to address multiplayer QoE fairness problem by designing better player bitrate adaptation algorithms [47, 54] and network-assisted bandwidth allocation schemes [31, 36, 58], there are still a lot of open questions in this space. For example, will the interaction among different classes of bitrate adaptation algorithms lead to instability? Is centralized, in-network or server-side control necessary to ensure multiplayer QoE fairness? How to design distributed control schemes with information exchange to achieve QoE fairness? We envision that this rich and broad problem space presents significant opportunities for control theory to provide insights to a real networking problem and to guide real system design. As a first step in this direction, we formalize the multiplayer QoE fairness problem and address a subset of the key questions.

We start from building a formal mathematical model of the multiplayer joint bandwidth allocation and bitrate adaptation problem, extending the single-player bitrate adaptation model from prior work [74, 75]. We first focus on the steady-state problem, and convert the multiplayer fairness problem as the stability analysis of an equilibrium of a non-linear dynamical system. We derive sufficent conditions under which multiple players with same/different bitrate adaptation policies can converge to QoE fairness with TCP-based bandwidth sharing at the bottleneck, and found that TCP-based network bandwidth sharing is not sufficient to ensure QoE fairness, confirming the observation of a measurement study [42] from a theory aspect. The result of the analysis calls for active, in-network support for better bandwidth allocation.

Given the recent development of smart routers such as Google OnHub router [10] and programmable OpenWrt [14], we envision that a router-based bandwidth allocation scheme is practical in the near future. While recent proposals of router-assisted schemes are based on steady-state utility maximization, we propose a non-linear MPC-based router-assisted bandwidth allocation algorithm that directly models players as close-loop dynamical systems. We evaluate the proposed strategy using trace-driven simulations and find that the router-assisted control outperforms existing steady-state solutions in both efficiency and fairness, by adaptively allocating more bandwidth to players which has high resolution and insufficient buffer level.

## 2.2 Background

We begin with a high-level overview of how HTTP-based adaptive video streaming works, before describing the key shortcomings of today's state-of-art solutions.

### 2.2.1 Single-player Video Streaming

Internet video technologies such as Microsoft SmoothStreaming [15], Apple's HLS [5], and Adobe's HDS [2] rely on *HTTP-based* adaptive streaming. This class of protocols is being standardized under the umbrella of Dynamic Adaptive Streaming over HTTP or DASH [18]. In DASH systems, each video consists of multiple segments or "chunks" (corresponding to a few seconds of play time) and each chunk is encoded at multiple *discrete* bitrates. The chunks from different bitrate streams are aligned so that the video player can switch to a different

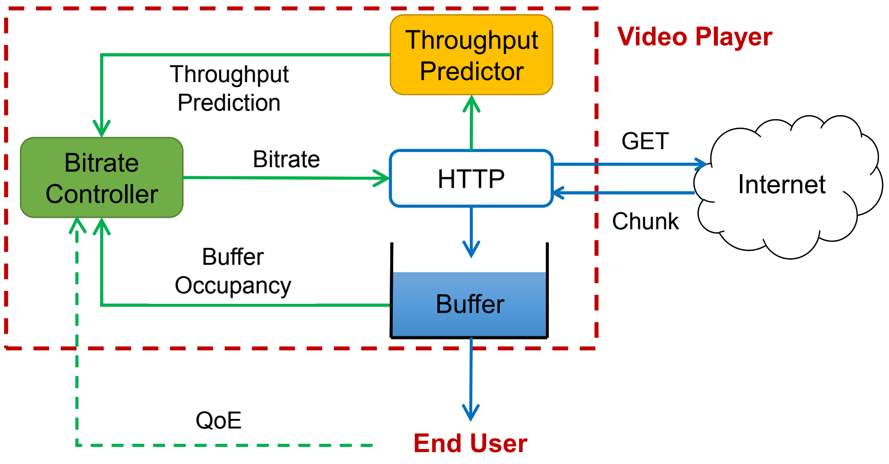


Figure 2.1: Abstract model of DASH players

bitrate if necessary at a chunk boundary. This approach has several pragmatic advantages over custom streaming protocols such as Real-Time Messaging Protocol (RTMP). The use of HTTP enables providers to seamlessly bypass middleboxes. Furthermore, it can use existing commodity CDN servers without requiring custom modifications. Finally, by making the server stateless, one can implement better application-layer resilience using multiple servers and CDNs [56, 57].

Figure 2.1 shows an abstract model of the adaptive video player. The player uses some inputs (e.g., buffer occupancy or estimates of the network throughput) in its decision logic to choose the bitrate level for the next chunk(s) to be downloaded. In making this decision, there are many potentially conflicting QoE considerations a player must account for: (1) minimizing rebuffering events where the playback buffer is empty and cannot render the video; (2) delivering as high a playback bitrate as possible within the throughput constraints; (3) minimizing startup delay so that the user does not quit while waiting for the video to load; and (4) keeping the playback as smooth as possible by avoiding frequent or large bitrate jumps [33, 50].

To see why these objectives are conflicting, let us consider two extreme solutions. A trivial solution to minimize rebuffering and the startup delay would be to always pick the lowest bitrate, but it conflicts with the goal of delivering high bitrate. Conversely, picking the highest available bitrate may lead to many rebuffering events. Similarly, the goal of maintaining a smooth playback may also conflict if the optimal choice to simultaneously minimize rebuffering and maximizing average bitrate is to switch bitrates for every chunk.

The focus of this dissertation is on *client-side* adaptation solutions. Other complementary work includes the use of server-side bitrate switching (e.g., [22, 52]), TCP changes to avoid bursts (e.g., [37]), and in-network throughput management and caching (e.g., [41, 60, 65]). We focus on the client-side problem for two key reasons. First, client-side solutions offer the most immediately deployable alternative in contrast to solutions that require in-network support (e.g., [41, 60, 65]), server-side software changes (e.g., [22, 52]), or modifications to lower-layer transport protocols (e.g., [37, 38, 40, 51, 55]). Second, the client is often in the best position to quickly detect performance issues and respond to dynamics. That said, we believe that the formal foundations and algorithms we develop can be equally applied to these other deployment scenarios.

Many measurement studies have shown the poor performance of state-of-art video players with respect to these QoE measures (e.g., [42, 47, 67]). These studies show that most problems are not artifacts of specific players but manifest across all state-of-art players such as SmoothStreaming [15], Netflix [12], Adobe OSMF [3], and Akamai HD [4]. For brevity we do not reproduce these results here but refer interested readers to prior work (e.g., [42, 47, 67]).

To alleviate these problems, there have been several recent proposals in the research literature (e.g., [22, 43, 47, 54, 69]). At a high level, these solutions can be roughly divided into two categories: (1) *rate-based algorithms* and (2) *buffer-based algorithms*. Video players with rate-based methods essentially pick the highest possible bitrate based on the estimated available throughput. However, as shown in prior work throughput estimation on top of HTTP suffers from significant biases [42], which leads to problems with traditional rate-based approaches. Some solutions try to work around these biases by either smoothing out throughput estimates [69] or choosing better scheduling strategies [47]. On the other hand, recent work makes a case for buffer-based algorithms [43]. Rather than using throughput estimates, this class of algorithms uses buffer occupancy as the feedback signal, and designs mechanisms to keep the buffer occupancy at a desired level, essentially discarding any available throughput information.

Despite the broad interest in this topic, we still need a principled understanding of bitrate adaptation algorithms. Each aforementioned solution offers point heuristics that work under specific (and implicit) environmental assumptions. While each approach seen in isolation has been shown to outperform commercial players, there is little effort to systematically compare how different *classes of algorithms* stack up against each other or which of these technical components are *critical*, or how *robust* these algorithms are across different operating regimes (e.g., throughput stability, buffer size, number of bitrate levels). Furthermore, many of these algorithms even fail to formally state what *objective* they seek to optimize making it harder to conduct a meaningful comparison.

Our first-order goal in this work is to bring some clarity to this space. Rather than design yet another point solution, we start by developing a first-principles approach via control theory to develop a general framework to reason about classes of algorithms. We use this control-theoretic "lens" to formally define the stochastic optimization that video bitrate adaptation algorithms try to solve.

### 2.2.2 Multi-player Video Streaming

While single-player bitrate adaptation algorithms have been well studied, they consider available bandwidth as a given stochastic variable and maximize QoE for a single player without considering the impact to other players. However, When multiple players share a bottleneck

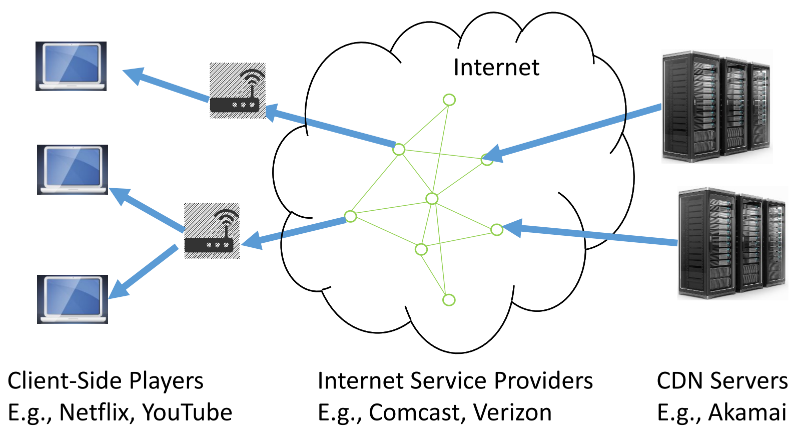


Figure 2.2: The Internet video delivery ecosystem

in the network, the efficiency and fairness of QoE across multiple adaptive video players become critical.

Note that multiplayer QoE fairness includes both fairness in *steady state* and *transient state*. For example, when a HDTV and a tablet share a bandwidth bottleneck in a home network, HDTV should ideally get more bandwidth in steady-state than the tablet as it needs higher-quality video to match the higher resolution. On the other hand, for example, a player with empty buffer is expected to obtain more bandwidth than another with full buffer sharing the same bottleneck, as it needs to quickly accumulate buffer so as to converge quickly to optimal bitrate and avoid rebuffering.

Different from single-player problem, the multiplayer QoE fairness can be affected by a broader range of factors. As such, we zoom out from the adaptive player model in Figure 2.1 and look at how the internet video delivery ecosystem impacts the multiplayer QoE fairness.

As shown in Figure 2.2, the Internet video delivery ecosystem consists of a variety of entities that has different control capabilities to optimize different objectives. Video source providers, such as Netflix and YouTube, own the client players and can design client-side bitrate control to optimize the user-perceived QoE; Content delivery networks (CDN), such as Akamai and Level3, place videos in CDN servers at the edge of the internet and assign players to best servers in a video session; Internet service providers (ISP), such as Comcast and Verizon, control the bandwidth available to CDN servers and client players according to agreement with users; Video quality optimizers, such as Conviva, employ a global view to provide centralized control of bitrate and CDN server selection for client players.

Given the diverse control capabilities in the internet video delivery system, there are several classes of solutions to achieve multiplayer QoE fairness: *player-side*, *in-network*, and *server-side* solutions.

Player-side solutions, such as FESTIVE [47] and PANDA [54], entail designing better bitrate adaptation algorithms for multiplayer QoE fairness. While only requiring player algorithm change and thus easy to deploy, player-side solutions do not alter bandwidth allocation in the network and can suffer from suboptimal bandwidth allocation schemes such as the unideal TCP effect [42] and interaction with uncooperative players and cross traffic [19].

In-network solutions, on the other hand, employ active bandwidth allocation in the network to achieve multiplayer QoE fairness. While bottleneck can occur anywhere in the network making such schemes difficult to deploy, there are several recent proposals in particular on router-based bandwidth allocation algorithms to optimize steady-state QoE fairness where router is the single bottleneck shared among players [31, 36, 58].

Alternatively, server-side solutions regard the server as a single point of control and allocate bandwidth to players [21]. However, the actual bandwidth bottleneck can occur in the network instead of server and the computation cost is high when the number of players is too large.

The broad problem space for multiplayer QoE fairness has posed a series of key research questions including:

1. What is the optimal approach and fundamental limitations of each class of solutions?

2. What is the fundamental tradeoff between different classes of solutions?

3. How to design the information exchange scheme to enable coordination of different entities in the video delivery ecosystems to achieve QoE fairness?

As a first step to tackle the broader problem, in this thesis we want to develop a principled framework and answer a subset of key questions so as to shed light on the broader problem space and provide useful insights for future work.

## 2.3 Single-Player Adaptive Video Streaming

First, we focus on the design of single-player bitrate adaptation algorithms.

### 2.3.1 Control-Theoretic Model

In this section, we develop a mathematical model of the HTTP video streaming process and formally define the bitrate adaptation problem. This model gives us a framework to compare and evaluate existing algorithms and serves as the foundation for potential improvements.

#### Video Streaming Model

We model a video as a set of consecutive *video segments* or *chunks*, $\mathcal{V} = \{1, 2, \cdots, K\}$, each of which contains $L$ seconds of video. Each chunk is encoded at different bitrates. Let $\mathcal{R}$ be the set of all available bitrate levels. The video player can choose to download chunk $k$ at bitrate $R_k \in \mathcal{R}$. Let $d_k(R_k)$ be the size of chunk $k$ encoded at bitrate $R_k$. In constant bitrate (CBR) case, $d_k(R_k) = L \times R_k$, while in variable bitrate (VBR) case the $d_k \sim R_k$ relationship can differ across chunks.

The higher bitrate is selected, the higher video quality is perceived by the user. Let $q(\cdot) : \mathcal{R} \to \mathbb{R}_+$ be a non-decreasing function which maps selected bitrate $R_k$ to video quality perceived by user $q(R_k)$. Note that $q(\cdot)$ may depend on the video-playing device as well as the content of the video. For example, while on HDTV 3Mbps and 1Mbps may lead to significant difference in user experience, the video quality in 3Mbps and 1Mbps may be similar on a mobile device; Also, improving the bitrate of "dynamic" chunks will result in more QoE gain than improving "static" chunks.

The video segments are downloaded into a *playback buffer*, which contains downloaded but as yet unviewed video. Let $B(t) \in [0, B_{max}]$ be the *buffer occupancy* at time $t$, i.e., the play time of the video left in the buffer. The *buffer size* $B_{max}$ depends on the policy of the service provider, as well as storage limitations on the player. A typical player buffer may hold few tens of seconds of video segments.

Figure 2.3 helps illustrate the conceptual operation of the video player. At time $t_k$, the video player starts to download chunk $k$. The download time for this chunk will be $d_k(R_k)/C_k$; i.e., it depends on the size of selected chunk with bitrate $R_k$, as well as average download speed $C_k$ experienced during this download process. Once chunk $k$ is completely downloaded, the video player waits for $\Delta t_k$ and starts to download the next chunk $k+1$ at time $t_{k+1}$. We assume that the waiting time $\Delta t_k$ is small and will not lead to rebuffering events. If we denote by $C_t$ the network throughput at time $t$, then we have:

$$t_{k+1} = t_k + \frac{d_k(R_k)}{C_k} + \Delta t_k \tag{2.1}$$

$$C_k = \frac{1}{t_{k+1} - t_k - \Delta t_k} \int_{t_k}^{t_{k+1}-\Delta t_k} C_t \, dt. \tag{2.2}$$

The buffer occupancy $B(t)$ evolves as the chunks are being downloaded and the video is being played. Specifically, the buffer occupancy increases by $L$ seconds after chunk $k$ is downloaded and decreases as the user watches the video.[1] Let $B_k = B(t_k)$ denote the buffer

[1]The "startup" phase will be slightly different as the player waits for some amount of buffer to build up before draining the buffer.

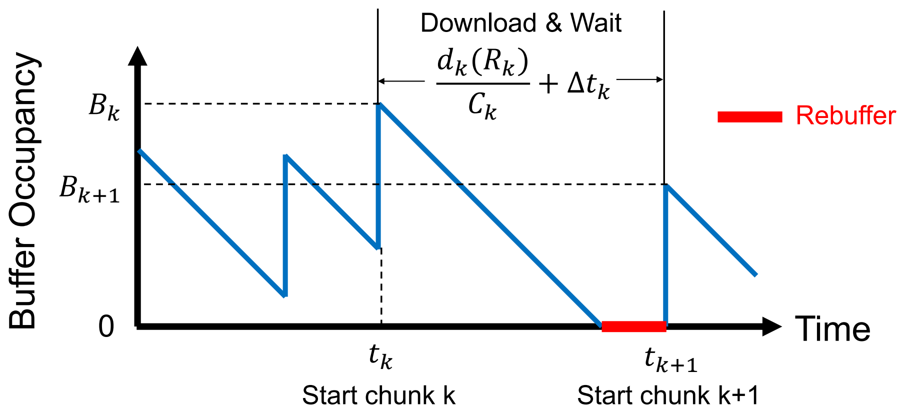


Figure 2.3: Illustration of buffer dynamics

occupancy when the player starts to download chunk $k$. The buffer dynamics can then be formulated as:

$$B_{k+1} = \left( \left( B_k - \frac{d_k(R_k)}{C_k} \right)_+ + L - \Delta t_k \right)_+ \tag{2.3}$$

Here, the notation $(x)_+ = \max\{x, 0\}$ ensures that the term can never be negative. Note that if $B_k < d_k(R_k)/C_k$, the buffer becomes empty while the video player is still downloading chunk $k$, leading to *rebuffer* events as shown in Figure 2.3.

The determination of waiting time $\Delta t_k$, also referred as *chunk scheduling* problem, is an equally interesting and important problem in improving fairness of multi-player video streaming [47]. However, in this dissertation we assume that the player immediately starts to download chunk $k+1$ as soon as chunk $k$ is downloaded. The one exception is when the buffer is full, the player waits for the buffer to reduce to a level which allows chunk $k$ to be appended. Formally,

$$\Delta t_k = \left( \left( B_k - \frac{d_k(R_k)}{C_k} \right)_+ + L - B_{max} \right)_+ \tag{2.4}$$

## QoE Maximization Problem

The ultimate goal of bitrate adaptation is to improve the QoE of users in order to achieve higher long-term user engagement [33]. Our goal is to provide a flexible QoE model rather than a fixed notion of QoE as this is an active area of research [23]. While users may differ in their specific QoE functions, we can enumerate the key elements of video QoE as:

1. *Average Video Quality:* The average per-chunk quality over all chunks: $\frac{1}{K}\sum_{k=1}^{K} q(R_k)$;
2. *Average Quality Variations*: This tracks the magnitude of the changes in the quality from one chunk to another: $\frac{1}{K-1}\sum_{k=1}^{K-1} |q(R_{k+1}) - q(R_k)|$;
3. *Rebuffer*: For each chunk $k$ rebuffering occurs if the download time $d_k(R_k)/C_k$ is higher than the playout buffer level when the chunk download started (i.e., $B_k$). Thus the *total rebuffer time* [2] is $\sum_{k=1}^{K}\left(\frac{d_k(R_k)}{C_k} - B_k\right)_+$.
4. *Startup Delay* $T_s$, assuming $T_s \ll B_{max}$.

As users may have different preferences on which of the four components is more important, we define the QoE of video segment 1 through $K$ by a weighted sum of the aforementioned components:

$$\begin{aligned} QoE_1^K = & \sum_{k=1}^{K} q(R_k) - \lambda \sum_{k=1}^{K-1} |q(R_{k+1}) - q(R_k)| \\ & - \mu \sum_{k=1}^{K}\left(\frac{d_k(R_k)}{C_k} - B_k\right)_+ - \mu_s T_s \end{aligned} \tag{2.5}$$

Here $\lambda, \mu, \mu_s$ are non-negative weighting parameters corresponding to video quality variations, rebuffering time and startup delay, respectively. A relatively small $\lambda$ indicates that the user is not particularly concerned about video quality variability; the large $\lambda$ is, the more effort is made to achieve smoother changes of video quality. A large $\mu$, relatively to the other

[2] Alternatively, one can also consider the *number of rebuffering events* formulated as $\sum_{k=1}^{K} \mathbf{1}\left(\frac{d_k(R_k)}{C_k} > B_k\right)$.

$$\max_{R_1,\cdots,R_K,T_s} \quad QoE_1^K \tag{2.6}$$

$$s.t. \quad t_{k+1} = t_k + \frac{d_k(R_k)}{C_k} + \Delta t_k, \tag{2.7}$$

$$C_k = \frac{1}{t_{k+1} - t_k - \Delta t_k} \int_{t_k}^{t_{k+1}-\Delta t_k} C_t \, dt, \tag{2.8}$$

$$B_{k+1} = \left( \left( B_k - \frac{d_k(R_k)}{C_k} \right)_+ + L - \Delta t_k \right)_+, \tag{2.9}$$

$$B_1 = T_s, \quad B_k \in [0, B_{max}] \tag{2.10}$$

$$R_k \in \mathcal{R}, \quad \forall k = 1, \cdots, K. \tag{2.11}$$

Figure 2.4: Formulation for QoE maximization ($QOE\_MAX_1^K$) subject to buffer and throughput dynamics.

parameters, indicates that a user is deeply concerned about rebuffering. In cases where users prefer low startup delay, we employ a large $\mu_s$.

In summary, this definition of QoE is quite general as it allows us to model varying user preferences on different contributing factors.

**QoE maximization problem:** We are now ready to formulate the problem of bitrate adaptation for QoE maximization as in Figure 2.4, denoted as $QOE\_MAX_1^K$. Given throughput trace $\{C_t, t \in [t_1, t_{K+1}]\}$ as input, the optimization provides the following as output: 1) bitrate decisions $R_1, \cdots, R_K$, and 2) startup time $T_s$.

Note that the problem $QOE\_MAX_1^K$ is formulated assuming the video playback has not started at the time of this optimization so the start-up delay $T_s$ is a decision variable. However, this QoE maximization can also take place during video playback at time $t_{k_0}$ when the next chunk to download is $k_0$ and the current buffer occupancy is $B_{k_0}$. In this case, we can drop the variable $T_s$ and denote the corresponding *steady state* problem as $QOE\_MAX\_STEADY_{k_0}^K$.

## Classes of Algorithms

In this section we characterize problem $QOE\_MAX_1^K$ and describe existing bitrate adaptation algorithms within this framework to understand how they relate to one another.

The problem in Figure 2.4 is a finite-horizon stochastic optimal control problem. The source of randomness is in the available throughput $C_t$. At time $t_k$ when the player chooses bitrate $R_k$, only the past throughput $\{C_t, t \leq t_k\}$ is available, while the future values $\{C_t, t > t_k\}$ are not known.

However, a *throughput predictor* can be used to obtain predictions defined as $\{\hat{C}_t, t > t_k\}$. Based on such prediction and on buffer occupancy information (which is instead known precisely), the *bitrate controller* selects bitrate of the next chunk $k$:

$$R_k = f\left(B_k, \{\hat{C}_t, t > t_k\}, \{R_i, i < k\}\right). \tag{2.12}$$

The design of effective throughput predictors is an interesting research direction in its own right [68]. In this dissertation, we focus on bitrate adaptation algorithms only and assume that predictors are given to us and are characterized in terms of their expected prediction errors. Namely, we focus on the design of $f(\cdot)$ and on the effect of the prediction error on the performance of the compared control algorithms. In the following sections, we will systematically evaluate how different algorithms perform with a state-of-art predictor under a variety of variability conditions.

Now, different adaptation algorithms essentially adopt different functions $f(\cdot)$. Specifically, two main categories of algorithms appear in the literature: *rate-based (RB)* and *buffer-based (BB)* algorithms. RB strategies essentially choose bitrate only based on throughput prediction, i.e.,

$$R_k = f\left(\{\hat{C}_t, t > t_k\}, \{R_i, i < k\}\right). \tag{2.13}$$

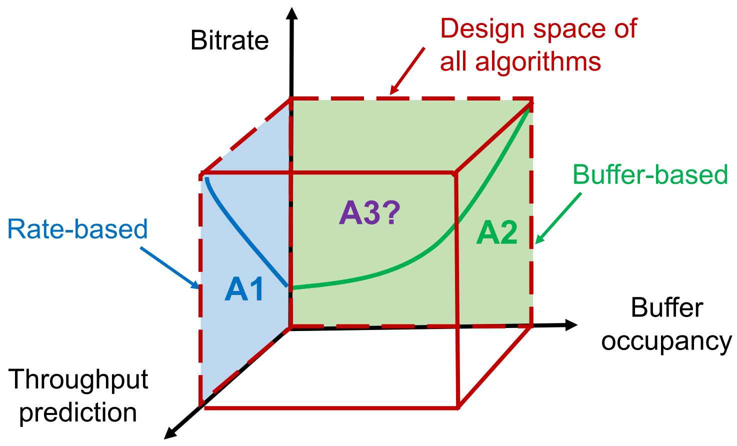


Figure 2.5: Design space of algorithms for the video adaptation problem: Most current approaches choose the bitrate as a function of only one variable; e.g., A1 is rate-based (RB) while A2 is buffer-based (BB).

For example, a typical RB strategy is to choose the maximum possible bitrate below the predicted throughput.

On the other hand, BB strategies advocate decision making based only on buffer occupancy, namely:

$$R_k = f\left(B_k, \{R_i, i < k\}\right), \tag{2.14}$$

while regarding throughput variations as unmodeled disturbances. For example, Huang et al., illustrate one roadmap for designing BB algorithms [43].

Note, however, that both classes of algorithms are discarding possibly useful information as shown in Figure 2.5. Consequently, both are in principle suboptimal. Ideally, we want to use both buffer occupancy and throughput prediction, thereby considering a broader design space of bitrate adaptation strategies, as shown in Eq (2.12) and algorithm A3 depicted in Figure 2.5.

### 2.3.2 Model Predictive Control Approach for Optimal Bitrate Adaptation

In this section, we make a case for a Model Predictive Control (MPC) approach for bitrate adaptation and describe a concrete MPC-based workflow that can optimally combine throughput prediction and buffer occupancy. We also develop a robust MPC approach that can better handle errors in throughput prediction under highly variable network conditions.

#### Why MPC?

First, we provide the intuition behind the choice of MPC in our setting. Note that we cannot claim that MPC is necessary or the optimal choice in the space of all possible control algorithms. Our goal is merely to argue that MPC is a natural fit for the bitrate adaptation problem.

**Strawman solutions:** As we saw before, bitrate adaptation is essentially a stochastic optimal control problem. In this respect, there are two candidate well-known control algorithms: (1) Proportional-integral-derivative (PID) control [34] and (2) Markov Decision Process (MDP) based control [25]. While PID is computationally simpler compared to MPC, it can only serve to stabilize the system and cannot explicitly optimize our QoE objective. In addition, PID control is designed to work in continuous time and state space and using it in a highly discrete system such as ours may result in performance degradation or instability [34]. Alternatively, with MDP we could consider formulating the throughput and buffer state transition as Markov processes, and find the optimal control policy using standard algorithms such as value iteration or policy iteration [25]. However, this has a strong assumption that throughput dynamics follow Markov processes and it is unclear if this holds in practice. We regard the potential use of MDP and analysis of the throughput dynamics as future work.

**Algorithm 1** Video adaptation workflow using MPC

```
1:  Initialize
2:  for $k = 1$ to $K$ do
3:      if player is in startup phase then
4:          $\hat{C}_{[t_k, t_{k+N}]}$= ThroughputPred($C_{[t_1, t_k]}$)
5:          $[R_k, T_s] = f_{mpc}^{st}\left(R_{k-1}, B_k, \hat{C}_{[t_k, t_{k+N}]}\right)$
6:          Start playback after $T_s$ seconds
7:      else if playback has started then
8:          $\hat{C}_{[t_k, t_{k+N}]}$= ThroughputPred($C_{[t_1, t_k]}$)
9:          $R_k = f_{mpc}\left(R_{k-1}, B_k, \hat{C}_{[t_k, t_{k+N}]}\right)$
10:     end if
11:     Download chunk $k$ with bitrate $R_k$, wait till finished
12: end for
```

**Case for MPC:** Ideally, given perfect knowledge of future throughput over the entire horizon of a video $[t_1, t_{K+1}]$, the optimal bitrate $R_1, \cdots, R_K$ and startup delay $T_s$ can be calculated in one shot by solving the optimization problem for the entire video $QOE\_MAX_1^K$. In practice, such perfect information is not available, making it difficult to find such optimal solutions using offline optimization.

While perfect information may not be available for the entire future, it is possible that reasonably accurate throughput prediction can be instead obtained for a short horizon to the future $[t_k, t_{k+N}]$ [68]. The intuition here is that network conditions are reasonably stable on short timescales and usually do not change drastically during a short horizon (tens of seconds) [68, 77]. Based on this insight, we can run a QoE optimization using the prediction in this horizon, apply the first bitrate $R_k$, and move the horizon forward to $[t_{k+1}, t_{k+N+1}]$. This scheme is known as model predictive control (MPC) or receding horizon control [28]. MPC algorithms are widely used in different domains, ranging from industrial control to navigation. The general benefits of MPC are in that MPC can utilize predictions to optimize a complex control objective online in a dynamical system under constraints.

**Basic MPC Algorithm**

Algorithm 1 shows a high-level overview of the workflow of MPC for bitrate adaptation. In our context, the algorithm essentially chooses bitrate $R_k$ by looking $N$ steps ahead (i.e., the moving horizon), and solves a specific QoE maximization problem (this depends on whether the player is in steady or startup phase) with throughput predictions $\{\hat{C}_t, t \in [t_k, t_{k+N}]\}$, or $\hat{C}_{[t_k,t_{k+N}]}$. The first bitrate $R_k$ is applied by using feedback information and the optimization process is iterated at each step $k$.

At iteration $k$, the player maintains a moving horizon from chunk $k$ to $k + N - 1$ and carries out the following three key steps, as shown in Algorithm 1.

1. *Predict*: Predict throughput $\hat{C}_{[t_k,t_{k+N}]}$ for the next $N$ chunks using some throughput predictor. Our goal in this dissertation is not to design a prediction mechanism but to rely on existing approaches. Naturally, improving the accuracy of this prediction will improve the gains achieved via MPC. That said, MPC can be extended to be robust to errors as we discuss below.

2. *Optimize*: This is the core of the MPC algorithm: Given the current buffer occupancy $B_k$, previous bitrate $R_{k-1}$ and throughput prediction $\hat{C}_{[t_k,t_{k+N}]}$, find optimal bitrate $R_k$. In *steady state*, $R_k = f_{mpc}\left(R_{k-1}, B_k, \hat{C}_{[t_k,t_{k+N}]}\right)$, implemented by solving $QOE\_MAX\_STEADY_k^{k+N-1}$. In the *start-up phase*, it also optimizes start-up time $T_s$ as $[R_k, T_s] = f_{mpc}^{st}\left(R_{k-1}, B_k, \hat{C}_{[t_k,t_{k+N}]}\right)$, implemented by solving $QOE\_MAX_k^{k+N-1}$. If we ignore practical details about computational overhead, we can simply use off-the-shelf solvers such as `CPLEX` to solve these discrete optimization problems. As we will see in Section 2.3.3, we do not need to explicitly solve the optimization problem within the video player in practice.

3. *Apply*: Start to download chunk $k$ with $R_k$ and move the horizon forward. If the player is in *start-up phase*, wait for $T_s$ before starting playback.

This workflow has several qualitative advantages compared with buffer-based (BB), rate-based (RB) as we discuss below. First, this MPC algorithm uses both throughput prediction and buffer information in a principled way. Second, compared to pure RB approaches, MPC smooths out prediction error at each step and is more robust to prediction errors. Specifically, by optimizing several chunks over a moving horizon, large prediction errors for one particular chunk will have lower impact on the performance. Third, MPC directly optimizes a formally defined QoE objective, while in RB and BB the tradeoff between different QoE factors is not clearly defined and therefore can only be addressed in an ad hoc qualitative manner.

**Robust MPC**

The basic MPC algorithm assumes the existence of an accurate throughput predictor. However, in certain severe network conditions, e.g., in cellular networks or in prime time when the Internet is congested, such accurate predictors may not be available. For example, if the predictor consistently overestimates the throughput, it may induce high rebuffering. To counteract the prediction error, we develop a *robust MPC* algorithm.

Robust MPC essentially optimizes the worst-case QoE assuming that the actual throughput can take any value in a *range* $[\underline{\hat{C}_t}, \overline{\hat{C}_t}]$ in contrast to a point estimate $\hat{C}_t$. Robust MPC entails solving the following optimization problem at time $t_k$ to get bitrate $R_k = f_{robustmpc}(R_{k-1}, B_k, [\underline{\hat{C}_t}, \overline{\hat{C}_t}])$:

$$\max_{R_k,\cdots,R_{k+N-1}} \min_{C_t \in [\underline{\hat{C}_t}, \overline{\hat{C}_t}]} QoE_k^{k+N-1} \tag{2.15}$$

$$s.t. \quad \text{Constraints (2.7) to (2.11)} \tag{2.16}$$

In general, it may be non-trivial to solve such a max-min robust optimization problem. In our specific case, however, we can prove that the worst case scenario takes place when the throughput is at its lower bound $C_t = \underline{\hat{C}_t}$. Thus, the implementation of robust MPC is straightforward. Instead of $\hat{C}_t$, we use the lowest possible $\underline{\hat{C}_t}$ as the input to the regular MPC QoE maximization problem. Formally,

**Theorem 1.** *The robust MPC controller is equivalent to the regular MPC taking the lower bound of throughput as input, namely,*

$$\begin{aligned} R_k &= f_{robustmpc}(R_{k-1}, B_k, [\underline{\hat{C}_t}, \overline{\hat{C}_t}]) \\ &= f_{mpc}(R_{k-1}, B_k, \underline{\hat{C}_t}) \end{aligned}$$

*Sketch.* Conceptually, QoE function $QoE(R, C)$ can be written as the sum of 3 terms ($g_1$: total video quality, $g_2$: total quality change, $g_3$: rebuffer time), in which only the rebuffer time term depends on throughput $C$. Thus,

$$\begin{aligned} &\max_R \min_{C \in [\underline{C}, \overline{C}]} \; QoE(R, C) \\ \equiv &\max_R \left( g_1(R) - \lambda \times g_2(R) - \max_{C \in [\underline{C}, \overline{C}]} \mu \times g_3(R, C) \right) \\ \equiv &\max_R \; QoE(R, \underline{C}) \end{aligned}$$

As any decrease of throughput $C$ will lead to longer rebuffer time, the minimum QoE is achieved at $C = \underline{C}$. $\square$

The one potential downside is that robust MPC is more conservative than regular MPC by always assuming the lowest throughput. The degree of conservativeness here naturally depends on how loose/tight the lower bound is. In practice, we use maximum prediction error over the past several chunks as bounds in our implementation and find that it works well in practice (discussed in Section 2.3.5).

### 2.3.3 Using MPC in Practice — FastMPC

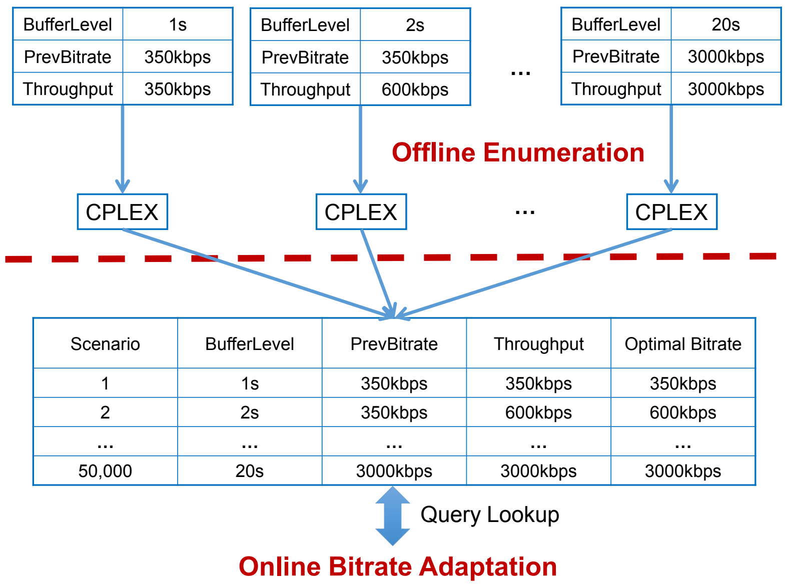
While rate-based and buffer-based algorithms need relatively minor computations, the challenge with MPC is that we need to solve a discrete optimization problem at each time step. There are two practical concerns here:

- *Computational overhead:* First, the high computational overhead of MPC is especially problematic for low-end mobile devices, which are projected to be the dominant video consumers going forward. Since the bitrate adaptation decision logic is called before the player starts to download each chunk, excessive delay in the bitrate adaptation logic will negatively affect the QoE of the player.
- *Deployment:* Since we do not have a closed-form or combinatorial solution for the QoE maximization problem, we will need to use a solver (e.g., `CPLEX` or `Gurobi`). However, it may not be possible for video players to be bundled with such solver capabilities; e.g., licensing issues may preclude distributing such software or it may require additional plugin or software installations which poses significant barriers to adoption [35].

From the above discussion, it is evident that the solution we develop should be *lightweight* and *combinatorial* (i.e., not solving a LP or ILP online). As such, in this section, we address these two key practical issues by developing a fast and low-overhead FastMPC design that does not require any explicit solver capabilities in the video player [70].

#### High-Level Idea of FastMPC

At a high level, FastMPC algorithms essentially follow a table enumeration approach. Here, we do an offline step of enumerating the state-space and solve each specific instance. Then, in the online step we just use these stored optimal control decisions mapped to the current operation conditions. That is, the algorithm will be reduced to a simple table lookup indexed



| Scenario | BufferLevel | PrevBitrate | Throughput | Optimal Bitrate |
|---|---|---|---|---|
| 1 | 1s | 350kbps | 350kbps | 350kbps |
| 2 | 2s | 350kbps | 600kbps | 600kbps |
| ... | ... | ... | ... | ... |
| 50,000 | 20s | 3000kbps | 3000kbps | 3000kbps |

Figure 2.6: "FastMPC" idea: We enumerate possible scenarios and create a table indexing the optimal decision for each scenario.

by the key value closest to the current state and the output of the lookup is the optimal solution for the selected configuration.

In our setting (Figure 2.6), the state-space is determined by the following dimensions: (1) current buffer level, (2) previous bitrates chosen, and (3) the predicted throughput for the next $N$ chunks (i.e., the planning horizon). Thus, FastMPC will entail enumerating potential scenarios capturing different values for each dimension and solving the optimization problems offline.

Unfortunately, directly using this idea will be very inefficient as we have a high dimensional state space. For instance, if we have 100 possible values for the buffer level, 10 possible bitrates, a horizon of size 5, and 1000 possible throughput values, there will be $10^{18}$ rows in the table![3] There are two obvious consequences of this large state space. First, it may not be practical to explicitly store the full table in the memory. Note that this is not just a hypothetical concern. If we need a practical implementation of this table lookup in the

[3]100 buffer levels $\times$ 10 bitrates $\times$ 1000 throughput 1 values $\times \cdots \times$ 1000 throughput 5 values $= 10^{18}$ entries.

`dash.js` player [1] it will mean very high memory footprint along with large startup delay as the table needs to be downloaded to the player module. Second, it will incur a non-trivial offline computation cost that may need to be rerun as the operating conditions change.

#### Optimizing FastMPC Performance

Next, we present two key optimizations to make the table enumeration approach tractable.

**Compaction via binning:** First, to address the offline exploration cost, our insight is that we may not need very fine-grained values for the buffer and the throughput levels. As a consequence these values may be suitably coarsened into aggregate bins. Moreover, with binning we do not need to explicitly store the row keys as these are directly computed from the bin row indices. The challenge here is to balance the granularity of binning and the loss of optimality in practice. In practice, we find that using 100 bins for buffer level and 100 bins for throughput predictions works well and yields near-optimal performance.

**Table compression:** Our second insight is that the decision table learned by the offline computation will have significant structure. Specifically, the optimal solutions for several similar scenarios will likely be the same. Thus, we can exploit this structure in conjunction with the binning strategy to explore a simple lossless compression strategy using a run-length encoding to store the decision vector. The optimal decision can then be retrieved online using binary search. In practice, we see that with compression the table occupies less than 60 kB with 100 bins for buffer levels, 100 bins for throughput predictions and 5 bitrate levels.

### 2.3.4 Implementation

In this section, we describe our implementation of the MPC approach in the `dash.js` framework. Our implementation is based on the `dash.js` master branch (v1.2.0 release) as it was the stable version at the time of development. We believe that our implementation can be easily adapted to future versions as we require minimal modifications ($\approx$ 800 lines of JavaScript). For more information on the source code and demo please visit our demo page [16].

**Choice of player:** Many prior adaptive bitrate players were prototyped using the Adobe OSMF framework [3, 13, 47] and this seemed a natural choice. However, our conversations with industry personnel revealed that almost all content providers are switching to HTML5-based players based on the MPEG-DASH standard [18] and thus OSMF (based on Flash and with decreasing market share) is unlikely to be a platform with real-world impact. Having chosen a DASH player, we qualitatively evaluated several implementations of the DASH standard (e.g., [8, 30, 61]). Unfortunately, these rely either on custom clients or niche video player platforms. Given these considerations, we chose the `dash.js` framework as it is the reference open-source implementation for the MPEG-DASH standard and is actively supported by leading industry participants [7]. We believe our prototype efforts will also inform the evolution of these standardization efforts. For instance, a key requirement for any control algorithms is to know the size (in bytes) of each video chunk, but the standard does not mandate the *manifest* to report chunk sizes, which may be a key shortcoming of the current specification.

`dash.js` **overview:** To understand our implementation and modifications, we begin with some brief background on the architecture of the `dash.js` player. The key components are highlighted in Figure 2.7.

At a high level, the `dash.js` implementation separates high-level video streaming functionalities from low-level specific DASH standard related components. As we are not particularly interested in standard-specific implementation, we leave the code unmodified and only focus on the adaptive streaming related functions.

The classes and functions that are key to bitrate adaptation and video streaming logic are as follows:

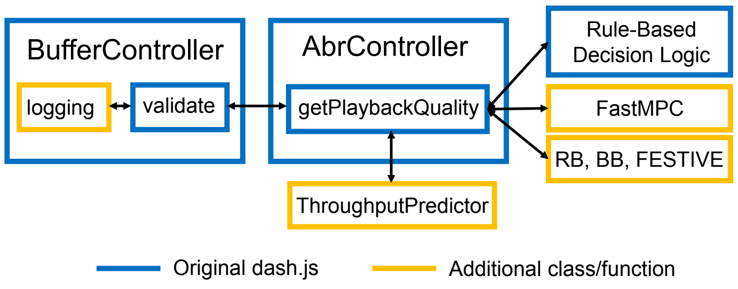


Figure 2.7: `dash.js` code structure and our modifications

- `BufferController`: This class provides functions to manage buffer levels of the player by requesting new segments and making bitrate change decisions. Specifically, function `validate` is periodically invoked and calls `getPlaybackQuality` function in `AbrController` class to find optimal bitrate. It also maintains a variable `bufferLevel` to record the current buffer occupancy of the player, which can be used for bitrate decisions.
- `AbrController`: This class contains the core bitrate adaptation logic. In the original `dash.js` implementation, a rule-based decision logic is employed to find the bitrate. Specifically, `DownloadRatioRule` selects bitrate based on the "download ratio" (play time of last chunk divided by its download time); On the other hand, `InsufficientBufferRule` chooses bitrate depending on whether the buffer level has reached a lower limit recently to avoid rebuffers. Priorities are assigned to each rule to resolve conflicts and make final bitrate decisions.

**Modifications and extensions:** We observed two implementation details in `dash.js` that were problematic. First, the code periodically calls the `validate` function to check the status of the buffer and call functions in `AbrController` to decide if the current bitrate should be changed. Note that this implies the bitrate decisions are not always made at chunk boundaries, which may lead to delay of execution of bitrate decisions, or even redownloading previous chunks. Second, the `dash.js` downloads multiple chunks in parallel even though chunks that are earlier in the video stream should ideally be prioritized.

To address these concerns, we changed the bitrate decision and chunk download process in `dash.js` code by making two key changes to `BufferController` class: 1) bitrate decisions are made at the start of each chunk, 2) chunk download is completely sequential, i.e., no concurrent downloads of multiple chunks are allowed. This allows a basic implementation framework which is consistent with our model and other proposed algorithms.

With these fixes, we implemented different bitrate adaptation algorithms (e.g., FastMPC, BB, RB) by replacing the original rule-based bitrate adaptation logic by our own implementation. The FastMPC implementation has a static table that is used to index control decisions. We also implemented a harmonic mean based throughput prediction scheme based on prior work [47], as well as additional logging functions in the `BufferController` class to record a complete log of the state of the player, including buffer level, bitrates, rebuffer time, predicted/actual throughput.

### 2.3.5 Evaluation

In this section, we compare our approach against existing rate- and buffer-based approaches using a combination of real player and simulation experiments. We also present microbenchmarks on the CPU and memory overhead of our FastMPC implementation.

#### Setup

We begin by describing key parameters: (1) throughput variability traces; (2) video-specific parameters; (3) configurations for various adaptation algorithms; and (4) definition of a normalized QoE metric that we use throughout this section.

**Throughput traces:** Our goal is to evaluate various bitrate adaptation approaches using realistic network variability conditions. Given the paucity of large-scale sustained throughput measurements over several tens of seconds, however, we use a combination of existing datasets and synthetic models:

1. *Broadband dataset (FCC) [9]:* The FCC dataset consists of more than 1 million sets of throughput measurements, where each set contains six data points each representing average throughput during a 5s interval. We extract throughput traces of the same server and client IP address and concatenate these to match the length of the video. For experiments we randomly pick 1000 of the concatenated traces whose average throughput is between 0 to 3Mbps, to avoid trivial cases where picking the maximum bitrate is always the optimal solution.
2. *Mobile dataset (HSDPA) [11]:* The HSDPA dataset consists of 30min of continuous 1s measurement of video streaming throughput of a moving device in Telenor's 3G/HSDPA mobile wireless network in Norway. We randomly pick 1000 throughput traces from the full dataset.
3. *Synthetic dataset:* Finally we also use a synthetic dataset to supplement the aforementioned datasets. The throughput is based on some hidden state $S_t \in \mathcal{S}$ modeling the number of users sharing a bottleneck link. The actual throughput $C_t$ follows a Gaussian distribution with mean $m_s$ and variance $\sigma_s^2$, given the value of hidden state $S_t = s$. We vary both the state transition probability matrix as well as the parameters $m_s$, $\sigma_s^2$ to generate traces.

Figure 2.8 shows the throughput characteristics of all three datasets. Among three datasets, throughput is the most stable in broadband network and the most variable in mobile network. In other words, the HSPDA dataset is a good stress test for our MPC approach that assumes the throughput is predictable on short timescales.

**Video parameters:** We use the "Envivio" video from DASH-264 JavaScript reference client test page [6] which is 260s long, consisting of 65 4s chunks. The video is encoded by H.264/MPEG-4 AVC codec in the following bitrate levels: $\mathcal{R} = \{350\text{kbps}, 600\text{kbps}, 1000\text{kbps}, 2000\text{kbps}, 3000\text{kbps}\}$. This is consistent with the requirement for YouTube video bitrate levels for 240p, 360p, 480p, 720p and 1080p respectively [17]. We set the buffer size

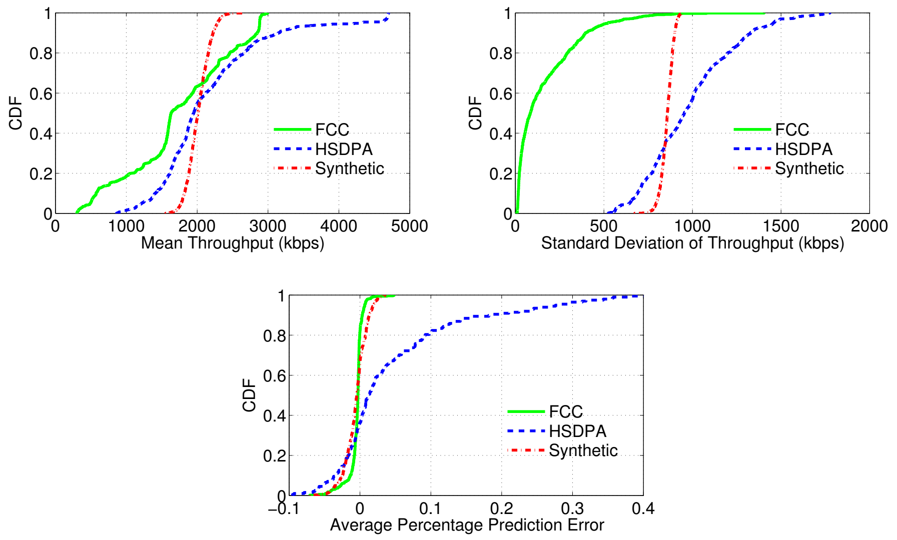
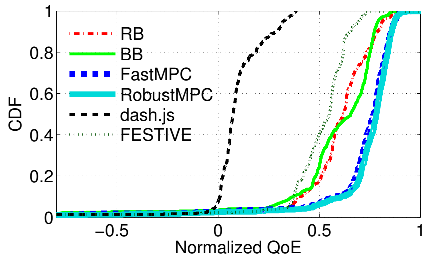


Figure 2.8: Characteristics of datasets

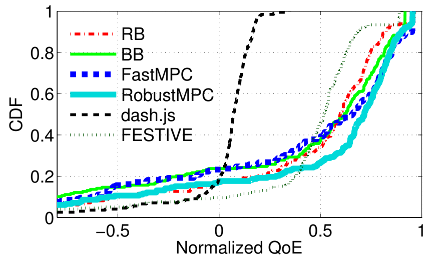
to $B_{max} = 30s$. We assume $q(\cdot)$ is an identity function. As a default QoE function, we use the weights $\lambda = 1$, $\mu = \mu_s = 3000$, meaning 1-sec rebuffer/start-up time receives the same penalty as reducing the bitrate of a chunk by 3000 kbps. We also run sensitivity experiments that vary the QoE weights.

**Adaptation algorithms:** Determining the optimal algorithm within each class is difficult as it involves optimizing over an infinite-dimensional functional space. To this end, we choose a widely adopted function form for each class of algorithms from prior work, and optimize the free parameters by empirical simulations based on a training dataset containing 100 throughput traces randomly picked across all datasets. We evaluate the following algorithms:

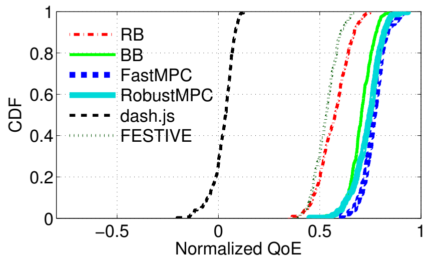
1. *RB:* The bitrate is picked as the maximum available bitrate which is less than $p = 1$ times throughput prediction using harmonic mean of past 5 chunks;
2. *BB:* We employ the function suggested by Huang et al [43], where bitrate $R_k$ is chosen to be the maximum available bitrate which is less than $r_k = f(B_k)$ with reservoir $r = 5s$ and cushion $c = 10s$.



(a) FCC dataset



(b) HSDPA dataset



(c) Synthetic dataset

Figure 2.9: Real experiment results with different throughput traces

3. *FastMPC:* We use a look-ahead horizon $h = 5$ with throughput predictions using harmonic mean of past 5 chunks; We use 100 bins for throughput prediction and 100 bins for buffer level. We also evaluate the exact MPC with perfect throughput prediction for the next 5 chunks in simulations (denoted as *MPC-OPT*).

4. *RobustMPC:* We assume that the throughput lower bound is $\underline{\hat{C}_t} = \hat{C}_t/(1 + err)$, where $\hat{C}_t$ is obtained using harmonic mean of past 5 chunks, while prediction error $err$ is the maximum absolute percentage error of the past 5 chunks.

5. `dash.js`*:* The original implementation adopts a rule-based bitrate decision logic as shown in Section 2.3.4. We keep the original bitrate adaptation logic unmodified, but disable the multi-chunk downloading and allow the bitrate to switch only at chunk boundaries.[4]

6. *FESTIVE [47]:* This rate-based algorithm balances both efficiency and stability, and incorporates fairness across players. We assume there is no wait time between consecutive

[4]This enables a consistent comparison of the algorithms rather than conflate it with other artifacts because of parallel downloads. We also tested the original `dash.js` without any modification, but its performance is worse than our modified version (not shown).

chunk downloads, and implement FESTIVE without the randomized chunk scheduling. Note that this does not negatively impact the player QoE in single player case. Specifically, FESTIVE calculates the efficiency score depending on $p = 1$ times throughput predictions using harmonic mean of past 5 chunks, as well as a stability score as a function of the bitrate switches in the past 5 chunks. The bitrate is chosen to minimize stability score plus $\alpha = 12$ times efficiency score.

**Throughput predictor:** Note that RB, *-MPC, and FESTIVE need a good throughput predictor. Developing good predictors for different scenarios is outside the scope of the dissertation, we refer interested readers to [68]. Building on insights from prior work, we use the harmonic mean of the observed throughput of the last 5 chunks because it is robust to outliers in per-chunk estimates [47].

**Normalized QoE metric:** We define a normalized QoE metric as follows. For a given throughput trace $\{C_t, t \in [t_1, t_{K+1}]\}$, the *offline optimal QoE*, denoted by $QoE(OPT)$, is the maximum QoE that can be achieved with perfect knowledge of future throughputs over the entire horizon. It can be calculated by solving problem $QOE\_MAX_1^K$ [5] and provides a theoretical upper bound of achievable QoE. On the other hand, a real *online* algorithm $A$ selects bitrate $R_k$ based on current throughput predictions $\{\hat{C}_t, t > t_k\}$ without knowing the entire future. We denote the online QoE achieved by algorithm $A$ by $QoE(A)$ and define *normalized QoE of A (n-QoE(A))* for an algorithm $A$ as: $n\text{-}QoE(A) = \frac{QoE(A)}{QoE(OPT)}$.

### Real Player Evaluation

First, we present emulations with the real player setup comparing our FastMPC approach against several prior approaches. Our basic experiment setup consists of two computers (Ubuntu 12.04 LTS) with a 100Mbps direct network connection emulating a video client

[5] To make it tractable to compute this offline optimal, we assume it can pick bitrates from a continuous range $[R_{min}, R_{max}]$.

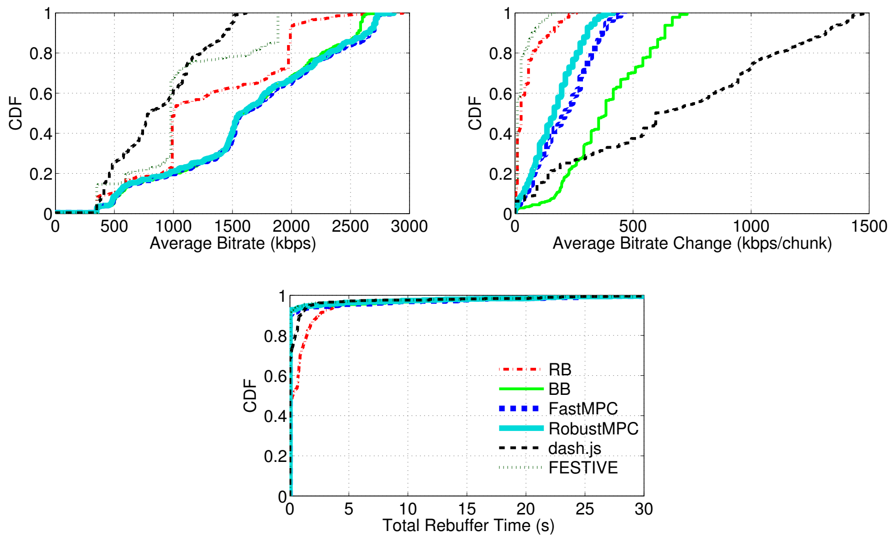


Figure 2.10: Detailed performance for FCC dataset

and server. The video client is a Google-Chrome web browser for linux (version 39) with V8 JavaScript engine while the video server is a simple HTTP server based on `node.js` (version 0.10.32). We use the linux `tc` tool to throttle the throughput of the link between two computers according to the throughput traces employed. We use Emulab [71] to carry out several such experiments in parallel.

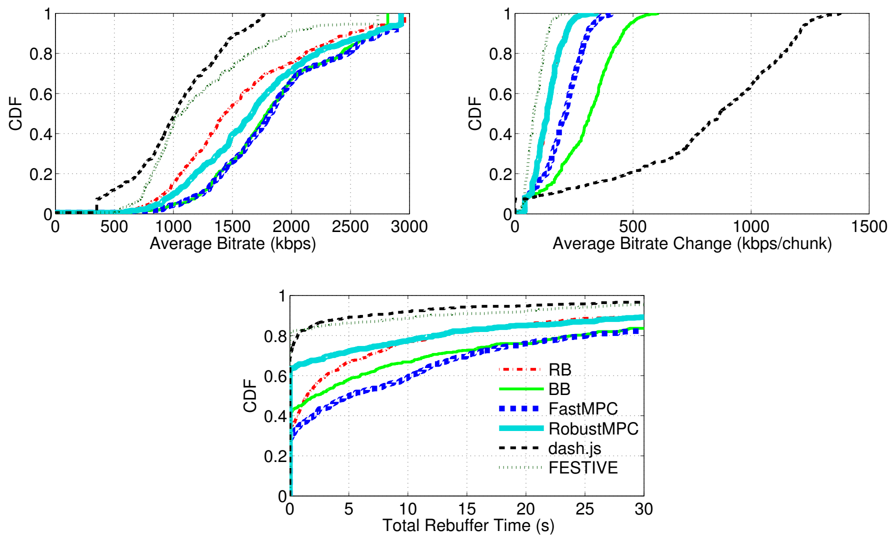


Figure 2.11: Detailed performance for HSDPA dataset

Figure 2.9 show the CDF of normalized QoE over the three sets of throughput traces. First, we see that existing algorithms achieve only 60-70% of optimal QoE confirming that there is still large room to improve video QoE. Second, RobustMPC outperforms non-MPC algorithms in all datasets with an improvement in median normalized QoE of 15%, 10%, and 5% in the FCC, HSDPA, and Synthetic datasets respectively. Third, we see significant improvement (60+% median normalized QoE) compared with the original `dash.js` player. Finally, we see that the basic FastMPC is more sensitive to prediction errors than RobustMPC. While there is no difference between Fast- and RobustMPC on FCC and Synthetic results, the difference is especially visible in the HSPDA result where regular FastMPC suffers and presents no gains versus RB and BB.

To better understand the impact of prediction error, Figure 2.8 shows the CDF of per-session average percentage prediction errors for the datasets. In FCC dataset, the average error of our harmonic mean throughput predictor is less than 5%, while in HSPDA dataset, the worst-case prediction error can be as high as 40%. We also observe that the predictor over-estimates the true throughput for more than 20% of the time in HSPDA dataset which leads to significant rebuffering. As such, inaccurate prediction can ruin the decision making of regular FastMPC, while RobustMPC is less affected as it incorporates prediction error to avoid choosing bitrate too aggressively when predictions are inaccurate.

The earlier normalized QoE result shows the aggregate combination of different QoE factors. Next, we zoom in on the individual quality factors to explain the QoE improvements in Figures 2.10 and 2.11. In the FCC dataset, all algorithms achieve similarly low rebuffer time as throughput is predictable. The performance difference essentially stems from reducing unnecessary bitrate switches. RobustMPC, FastMPC and BB achieve similar average bitrates, but RobustMPC uses fewer bitrate switches. In the HSPDA result, rebuffer time becomes a more important issue. While FastMPC achieves similar average bitrate and fewer switches comparing to BB, it suffers from large rebuffer time. On the other hand, RobustMPC achieves significant less rebuffer time but at a slightly lower average bitrate: Zero rebuffer in 65% of all cases, versus 40% for BB and FastMPC. As a result, RobustMPC still outperforms other algorithms in overall QoE.

Doing a cross-dataset analysis, we see that the tail distributions of the overall QoE show different characteristics. In the FCC result, only 1% users experience normalized QoE ¡0 while in HSPDA this occurs in 10% of all cases.[6] Again, the main reason is that the high variability of mobile network induces long rebuffering which affects the overall QoE.

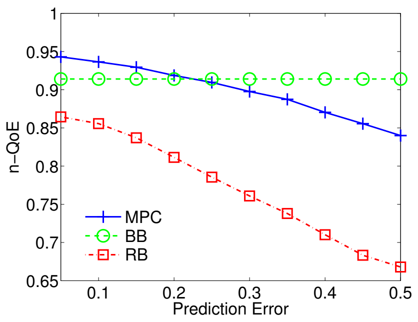
Finally, even though FESTIVE is a rate-based algorithm, it performs slightly worse than regular RB in our datasets because it puts a higher weight on stability and switches up bitrate slowly even when the available throughput is increasing.[7] On the other hand, the `dash.js` heuristic rule-based adaptation achieves low rebuffer time, but incurs many unnecessary switches. Thus, its overall QoE is significantly worse than all algorithms.

## Sensitivity Analysis

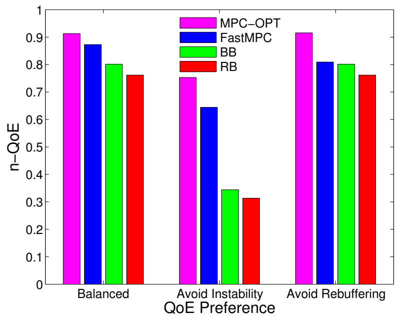
For sensitivity analysis we evaluate different algorithms using a custom simulation framework. As before, the simulation takes as input a throughput trace and models the video download/playback process and the buffer dynamics. At time $t_k$ when the bitrate of chunk $k$ is needed, the simulation calls the bitrate controller embedded with different algorithms to get $R_k$. Using this framework, we study the sensitivity of the approaches to key factors such as: (1) prediction error, (2) choice of QoE function, (3) playout buffer size, (4) number of bitrate levels, and (5) startup delay.

**Throughput prediction:** Here, we want to study the impact of prediction error of general predictors rather than analyze a particular one (e.g., harmonic mean). To this end, we use the average error level to characterize the performance of a throughput predictor and model the

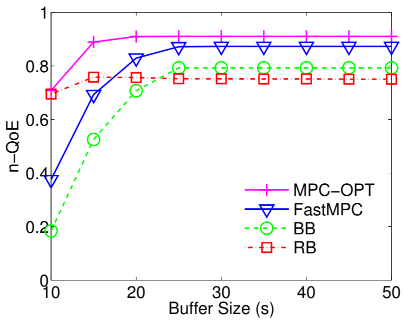
[6]The QoE can be negative when rebuffer time is too long or there are too many switches.
[7]This is not a flaw, but a deliberate choice for achieving multi-player fairness [47].



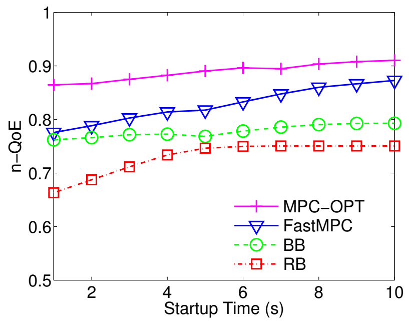
(a) Prediction error



(b) QoE preferences



(c) Buffer size



(d) Startup time

Figure 2.12: Sensitivity analysis vs. operating conditions

prediction output as being a combination of the true throughput with added random noise according to the average error level. Figure 2.12a shows how the throughput prediction errors influence the performance of bitrate adaptation algorithms. As expected, BB is unaffected as it does not use any throughput information. When throughput predictions are accurate, MPC has larger advantage over BB algorithms. As prediction error grows beyond 25%, MPC can be even worse than BB. This suggests that if the actual prediction error is very large, then the video player should drop RB or MPC and use pure BB algorithms. In contrast with regular MPC, robust MPC is less affected by prediction error as it takes into possible error into account and maximizes the worst case QoE.

**Users' QoE preferences:** We compared the performance of the algorithms under 3 sets of QoE weights, "Balanced" ($\lambda = 1, \mu = \mu_s = 3000$), "Avoid Instability" ($\lambda = 3, \mu = \mu_s = 3000$), "Avoid Rebuffering" ($\lambda = 1, \mu = \mu_s = 6000$). As shown in Figure 2.12b, as users put more penalty weights to bitrate instability, the MPC algorithms show more advantage over RB and BB. This is because MPC algorithms explicitly model the bitrate vs. bitrate instability tradeoff in the QoE function, while RB and BB do so in ad-hoc ways. However, when rebuffering time is a more important factor, BB algorithms perform similarly with FastMPC algorithms because of two key reasons. First, BB algorithms keep a minimum buffer level so that the player has a better chance surviving low throughput with less/no rebuffering time. Second, while MPC algorithms do a good job with perfect throughput prediction, they can suffer from long rebuffering time since harmonic mean predictor is imperfect. As such, MPC can be improved by maintaining a minimum buffer level and employing a more accurate predictor.

**Buffer size and startup delay:** Figure 2.12c analyzes the impact of playout buffer size. First, when buffer size is small (¡25s in play time), increasing buffer size improves the performance of all algorithms. A larger buffer protects the player against rebuffering events and also provides more degrees of freedom to optimize performance. As buffer size reaches a certain level (25s of play time), the performances of all algorithms stay constant even buffer size is further increased. Finally, RB is the least affected by buffer size because it does not consider buffer level in its decision logic.

While our approach optimizes startup delay automatically, we analyze how overall QoE (except the startup delay term) is affected if the startup delay is fixed. As shown in Figure 2.12d, as startup time increases, the performance of all algorithms improves, as the player accumulates more video in the buffer at the start-up phase making it easier to manage rebuffering events.

**Bitrate levels:** We also study how number of bitrate levels influences the performance (not shown). With BB and MPC, we can achieve better performance using finer-grained set of bitrate levels. With RB, however, the performance of RB first improves as we add more bitrate levels, but decreases when there are too many bitrate levels. The reason is that RB starts changing bitrate more frequently, leading to increased bitrate instability. One caveat with MPC is that finer-grained bitrate levels also require more discretization levels for the FastMPC implementation. Understanding this tradeoff is an interesting direction for future work.

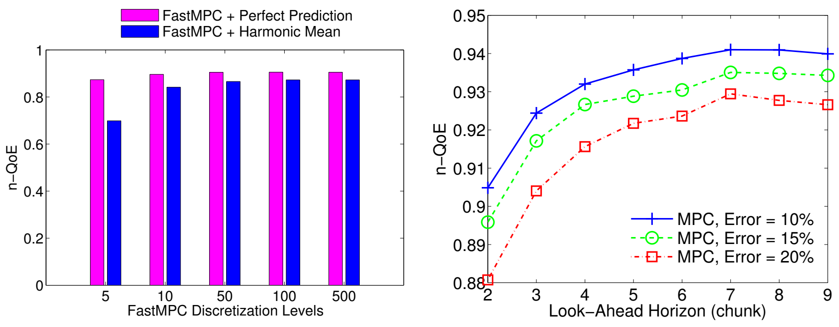
### MPC Configuration and Overhead

**Overhead:** As discussed earlier, FastMPC might increase player overhead relative to BB and RB style algorithms. We compare the CPU and memory usage of our implementation of FastMPC, BB, and RB algorithm with the default `dash.js` player. We find that FastMPC, BB, and RB all consume similar amount of CPU, while FastMPC uses only 60 kB more memory (not shown).

**FastMPC discretization:** Recall that the number of discretization levels is an important design parameter for FastMPC. More discretization levels increase FastMPC performance but require more player memory and may also increase startup delay. We study this performance vs. overhead tradeoff in Figure 2.13a and Table 2.1. From Figure 2.13a, we see that more discretization levels imply larger performance gains for FastMPC but the improvement shows diminishing return; e.g., FastMPC achieves 90% of optimal QoE with 100 levels while this drops to 70% if there are only 5 levels. Second, the gain vs. discretization level also has some dependency on the throughput predictor especially with very coarse discretization. Table 2.1 shows that while the memory overhead increases with more levels, the simple compression scheme we discussed earlier can reduce the memory overhead especially when number of

| Discretization levels | Extra JavaScript code size | |
|---|---|---|
| | Full table | Run length coding |
| 50 | 25.0 kB | 19.1 kB |
| 100 | 100 kB | 56.4 kB |
| 200 | 400 kB | 141 kB |
| 500 | 2.50 MB | 451 kB |

Table 2.1: FastMPC table size



(a) Discretization (b) Look-ahead horizon

Figure 2.13: MPC configuration parameters

levels is large. For instance, with 100 levels the compression rate is 0.5 while with 500 levels it can reduce the table size by 82%. Even with 500 levels, the table size is quite reasonably low.

**Look-ahead horizon:** Figure 2.13b shows how planning horizon impacts the performance of MPC algorithms. As the look-ahead horizon increases, MPC performances grow and stay stable since more information of future throughput is taken into account. However, as we look further into the future, prediction accuracy can reduce. The performance of MPC can even drop if the horizon is too large.

### Summary of Results

Our main findings are summarized as follows:

1. RobustMPC outperforms existing algorithms in both broadband (FCC) and cellular (HSDPA) datasets, while regular FastMPC does not show advantage in cellular network due to high throughput instability;
2. Our implementation of FastMPC algorithm incurs very low overhead: near-zero CPU overhead and 60 kB increase in memory usage compared to original `dash.js`;

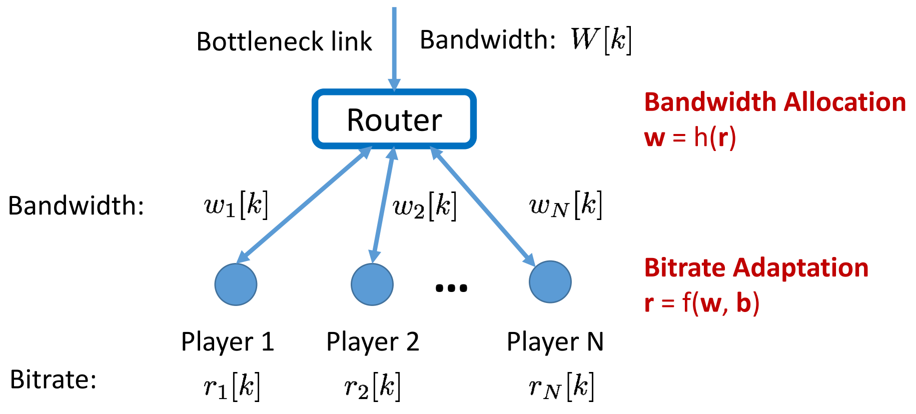


Figure 2.14: Modeling multiplayer joint bandwidth allocation and bitrate adaptation problem

3. Sensitivity analysis shows that FastMPC has advantages over BB and RB in wide parameter ranges. However, there is still room for improvement by increasing FastMPC discretization granularity and employing more accurate throughput predictors.

## 2.4 Multi-Player Adaptive Video Streaming

Next, we switch our focus to multiplayer adaptive video streaming problem and investigate how to jointly design better bandwidth allocation and bitrate adaptation schemes.

### 2.4.1 Modeling

In this section, we develop a mathematical model for multiplayer HTTP-based adaptive video streaming. Figure 2.14 provides an overview of the model. Note that while this is extending the single-player video streaming model discussed in previous section 2.3.1, we adopt synchronized time steps for all players in multi-player scenario.

**Video streaming model:** We consider a discrete time model with time horizon $\mathcal{K} = \{1, \cdots, K\}$ with a sampling period $\Delta T$. Let us consider a set of $N$ video players $\mathcal{P}$ sharing a single bottleneck link with bandwidth $W[k]$ at time $k$. Let $w_i[k] \in \mathbb{R}_+$ be the available bandwidth to the player $i$ at the time $k$, we have:

$$\sum_{i \in \mathcal{P}} w_i[k] \leq W[k], \quad \forall k \in \mathcal{K} \tag{2.17}$$

We assume this link is the only bottleneck along the Internet path from the video players to the servers.

Each video player streams video from some video server on the Internet via HTTP. The video is encoded in a set of bitrate levels $\mathcal{R}$. When downloading video, player $i \in \mathcal{P}$ is able to choose the bitrate $r_i[k] \in \mathcal{R}$ of the video at each time step $k$. In constant bitrate encoding, $r_i \times t$ bits of data need to be downloaded to get the video with $t$ seconds of play time.

Each player has a buffer to store downloaded yet unplayed video. Let $b_i[k] \in [0, \overline{B}_i]$ be the buffer level at the beginning of time step $k$, namely, the amount of play time of the video in the buffer. The buffer accumulates as new video is being downloaded, and drains as video is played out to users. The buffer dynamics of the player $i$ is formulated as follows:

$$b_i[k+1] = b_i[k] - \Delta T + \frac{w_i[k]\Delta T}{r_i[k]} \tag{2.18}$$

**QoE objective:** The objective of the adaptive video players is to maximize the quality-of-experience (QoE) of users, which is modeled as a linear combination of the following factors: 1) average video quality, 2) average quality change, 3) total rebuffer time and 4) startup delay (see previous section 2.3.1 for details). For simplicity, in this dissertation we enforce that there are no rebuffering events, and we only consider the case where all the players have started playback. As such, the QoE utility function $U_i : \mathcal{R} \times \mathbb{R}_+ \times \mathbb{R}_+ \to \mathbb{R}$ of player $i$ is the formulated as the average QoE of video downloaded over the entire time horizon:

$$U_i = \frac{\sum_{k=1}^{K} \frac{w_i[k]}{r_i[k]} U_i^P[k]}{\sum_{k=1}^{K} \frac{w_i[k]}{r_i[k]}} \tag{2.19}$$

where $U_i^P[k]$ is the QoE of the video downloaded in time $k$:

$$U_i^P[k] = q_i(r_i[k]) - \mu_i \left| q_i(r_i[k]) - q_i(r_i[k-1]) \right| \tag{2.20}$$

Note that $q_i : \mathcal{R} \to \mathbb{R}$ is the function that maps bitrate to the video quality perceived by users. We assume $q_i(\cdot)$ to be positive, increasing and concave to model the diminishing return property. $\mu_i$ is the parameter that defines the trade-off between high average quality and less quality changes. The larger $\mu_i$ is, the more reluctant the user $i$ is to change the video quality.

**QoE fairness:** Going from single player to multiplayer video streaming, a natural objective function would be the sum of utilities (QoE) of all users, also known as *social welfare* or *efficiency*, i.e., $\sum_{i \in \mathcal{P}} U_i$. However, in the context of multiplayer video streaming, QoE fairness among players becomes a critical issue as each player usually serves a different user yet they share the same bottleneck resource. As such, we consider the QoE fairness $F(U_1, \cdots, U_N)$ as the objective, where $F : \Pi_{i \in \mathcal{P}} \mathcal{U}_i \to \mathbb{R}$ is a general fairness measure [53]. Specifically, we consider a class of fairness measures known as $\alpha$-fairness [59], where:

$$F_\alpha(\mathbf{U}) = \begin{cases} \sum_{i \in \mathcal{P}} \frac{U_i^{1-\alpha}}{1-\alpha} & \alpha \geq 0, \alpha \neq 1 \\ \sum_{i \in \mathcal{P}} \log U_i & \alpha = 1 \end{cases} \tag{2.21}$$

Note that $\alpha$-fairness is a general fairness measure that satisfies axiom 1,2,3,5 from [53]. If $\alpha = 1$, $\alpha$-fairness becomes *proportional fairness*; if $\alpha \to \infty$, it becomes *max-min fairness.*

**Multiplayer QoE maximization problem:** Now we are ready to formulate the multiplayer QoE maximization problem where optimal bitrates $(\mathbf{r}[k], k \in \mathcal{K})$ and bandwidth $(\mathbf{w}[k], k \in \mathcal{K})$ of players are decided to maximize some QoE fairness measure $F(\mathbf{U})$, given the capacity of the bottleneck link, $(W[k], k \in \mathcal{K})$:

$$\begin{aligned}
\max \quad & F(U_1, \cdots, U_N) && (2.22)\\
\text{over} \quad & \mathbf{r}[k], \mathbf{w}[k] \quad \text{given } W[k], k \in \mathcal{K} && (2.23)\\
s.t. \quad & \sum_{i \in \mathcal{P}} w_i[k] = W[k], \quad \forall k \in \mathcal{K} && (2.24)\\
& b_i[k+1] = b_i[k] - \Delta T + \frac{w_i[k]\Delta T}{r_i[k]}, && (2.25)\\
& \qquad \forall i \in \mathcal{P}, k = 1, \cdots, K &&\\
& \underline{B}_i \le b_i[k] \le \overline{B}_i, \quad \forall i \in \mathcal{P}, k \in \mathcal{K} && (2.26)\\
& w_i[k] \ge 0, r_i[k] \in \mathcal{R} \quad \forall i \in \mathcal{P}, k \in \mathcal{K} && (2.27)
\end{aligned}$$

Ideally, a centralized controller can decide both the bitrate $\mathbf{r}$ and the bandwidth $\mathbf{w}$ for all players to achieve QoE fairness, given the complete information of the system. However, the current practice can be interpreted as a distributed way to solve the problem by primal decomposition with no explicit message passing between players and router: Each player $i$ decides the bitrate of itself according to some *bitrate adaptation policy* $r_i[k+1] = f(w_i[k], b_i[k+1])$, while the bottleneck link (conceptually) decides how to allocate available bandwidth according to some *bandwidth allocation policy* $\mathbf{w}[k] = h(\mathbf{r}[k], \mathbf{b}[k])$. The design of optimal distributed solution is to find optimal $(h, f)$ pairs, i.e., $(h^*, f^*)$. Next, we discuss respectively the design of $h$ and $f$.

**Bandwidth allocation policies:** Given that the players in $\mathcal{P}$ shares a bottleneck link with total bandwidth $W[k]$, i.e., $\sum_{i \in \mathcal{P}} w_i[k] = W[k]$. A *bandwidth allocation* policy $h : \mathbb{R}^n \to \mathbb{R}^n$ is a function that maps bitrates $\mathbf{r}[k]$ to bandwidth allocation vector $\mathbf{w}[k]$. Let $h_i : \mathbb{R}^n \to \mathbb{R}$

be the function that maps $\mathbf{r}[k]$ to $w_i[k]$.

$$\mathbf{w}[k] = h(\mathbf{r}[k]) \tag{2.28}$$

Under ideal TCP, all players get the equal share of the total bandwidth, i.e., $w_1[k] = \cdots = w_N[k] = W[k]/N$,. However, in practice, TCP is not ideal in the sense that players with larger bitrate gets larger share of the bandwidth due to the discrete effects [42]. We have the following assumptions of the bandwidth allocation function under unideal TCP according to measurement data in [42]:

**Assumption 1.** *Under non-ideal TCP, the bandwidth allocation policy $h(\cdot)$ has the following properties:*

1. *If $r_i = r_j$, $h_i(\mathbf{r}) = h_j(\mathbf{r})$;*
2. *If $r_i > r_j$, $h_i(\mathbf{r}) > h_j(\mathbf{r})$;*
3. *$\frac{\partial h_i(\mathbf{r})}{\partial r_i} > 0$, $\frac{\partial h_i(\mathbf{r})}{\partial r_j} < 0$, $i \neq j$;*
4. *$\lim_{r_i \to \infty} h_i(\mathbf{r}) < W$, $\lim_{r_i \to 0} h_i(\mathbf{r}) > 0$;*
5. *$h(\cdot)$ is symmetric over $\mathbf{r}$ (does not depend on order of players).*

**Lemma 1.** *The function $h(\cdot)$ has $1 + kn$ fixed points, where $k \in \mathbb{N}$.*

**Bitrate adaptation policies:** Bitrate adaptation policy of player $i$, $f_i(\cdot)$, maps available bandwidth $w_i[k]$ and buffer level $b_i[k]$ to bitrate to choose $r_i[k]$ so as to maximize the QoE of the player. Bitrate adaptation policies have been widely studied by both in academia and in industry, and each video streaming service has its own adaptation policy. To make decisions on what bitrate to choose, there are two classes of algorithms: rate-based (RB) or buffer-based (BB) controllers.

In a rate-based policy $RB(f_i)$, $r_i[k] = f_i(w_i[k-1])$, where $f_i : \mathbb{R}_+ \to \mathcal{R}$ is an increasing function. We consider a special case $LRB(\alpha)$ where $f_i$ is an affine function $r_i[k] = \alpha w_i[k-1]$.

In a buffer-based policy $BB(f_i)$, $r_i[k] = f_i(b_i[k])$, where $f_i : \mathbb{R}_+ \to \mathcal{R}$ is an increasing function. We also consider the special case $LBB(\alpha, \beta)$ of an affine $f$ function $r_i[k] = \alpha b_i[k] + \beta$.

Note that both RB and BB policies can be regarded as heuristic algorithms to maximize QoE which may lead to sub-optimal solution. However, it is still of great interest to study these policies as they are currently widely deployed in the real-world players, such as Netflix or YouTube.

### 2.4.2 Analysis of Fairness in Steady State

**QoE fairness in the steady state:** Note that an interesting special case of the multiplayer problem is when the system is in steady state, where the video quality and bandwidth of all players stay unchanged. Formally, we have the following definition:

**Definition 1.** *Given fixed total available bandwidth $W$, the multiplayer video streaming system is in steady state $(\mathbf{r}_0, \mathbf{w}_0)$ if for each player $i \in \mathcal{P}$:*

1. *Bitrate and bandwidth stay unchanged, i.e., $r_i[k] = r_{0i}$, $w_i[k] = w_{0i}$, $\forall k \in \mathcal{K}$;*
2. *Buffer level is non-decreasing, i.e., $b_i[k+1] \geq b_i[k]$, $\forall k \in \mathcal{K}$.*

Removing the inter-temporal constraints (2.25) and inter-temporal component in the objective function (2.20), we get the multiplayer QoE fairness problem in steady state where optimal solution is denoted as $(\mathbf{r}_0^*, \mathbf{w}_0^*)$:

$$\begin{aligned}
\max \quad & f\left(q_1(r_1), \cdots, q_N(r_N)\right) && (2.29)\\
\text{over} \quad & \mathbf{r}, \mathbf{w} \quad \text{given } W && (2.30)\\
s.t. \quad & \sum_{i \in \mathcal{P}} w_i = W, && (2.31)\\
& r_i \leq w_i, \quad \forall i \in \mathcal{P} && (2.32)\\
& w_i \geq 0, r_i \in \mathcal{R}, \quad \forall i \in \mathcal{P} && (2.33)
\end{aligned}$$

Note that this problem is convex given that $\mathcal{R} = [\underline{R}, \overline{R}]$, and in the case that all players share the same $q_i = q$, the optimal solution is $(\mathbf{r}_0^*, \mathbf{w}_0^*) : r_{0i} = w_{0i} = W/N$.

**Fairness of homogeneous RB players:** We first consider the simplest case where all players are using the same rate-based algorithm.

**Theorem 1.** *If all players adopt $RB(f)$ bitrate adaptation policies, the following statements are true:*

1. *$(\mathbf{r}_0, \mathbf{w}_0) : \ r_{i0} = f\left(\frac{w}{n}\right), w_{i0} = \frac{w}{n}$ is an equilibrium;*
2. *If $h \circ f$ is a contractive mapping, $(\mathbf{r}_0, \mathbf{w}_0)$ is globally asymptotically stable;*
3. *If $h \circ f$ is a expansive mapping, $(\mathbf{r}_0, \mathbf{w}_0)$ is unstable;*

*Proof.* Combining rate-based policies with buffer dynamics, we get:

$$r_i[k+1] = f\left(h_i(\mathbf{r}[k])\right), \quad \forall i \in \mathcal{P}$$

Consider the following Lyapunov function:

$$V(\mathbf{r}) = \sum_{i \in \mathcal{P}} \left(h_i(\mathbf{r}) - h_i(\mathbf{r}_0)\right)^2$$

which satisfies $V(\mathbf{r}) > 0, \forall \mathbf{r} \neq \mathbf{r}_0$, $V(\mathbf{r}_0) = 0$.

$$\begin{aligned}
\Delta V(\mathbf{r}) &= V(\mathbf{r}[k+1]) - V(\mathbf{r}[k]) = \cdots \\
&= \|\mathbf{w}[k+1]\|_2^2 - \|\mathbf{w}[k]\|_2^2 \\
&= \|h(f(\mathbf{w}[k]))\|_2^2 - \|\mathbf{w}[k]\|_2^2
\end{aligned}$$

If $h \circ f$ is a contractive mapping, $\Delta V(\mathbf{r}) < 0, \forall \mathbf{r} \neq \mathbf{r}_0$, and $\Delta V(\mathbf{r}_0) = 0$, the equilibrium is globally asymptotically stable. On the other hand, if $h \circ f$ is expansive, $\Delta V(\mathbf{r}) > 0, \forall \mathbf{r} \neq \mathbf{r}_0$, and $\Delta V(\mathbf{r}_0) = 0$, the equilibrium is unstable. □

**Fairness of homogeneous BB players:** We consider the case where all players adopt the same buffer-based bitrate adaptation policy and have the same QoE function.

**Lemma 2.** *If all players adopts buffer-based bitrate adaptation policy, $(\mathbf{r}_0, \mathbf{w}_0)$ is an equilibrium if and only if:*

1. $\mathbf{r}_0 = \mathbf{w}_0$;
2. $h(\mathbf{r}_0) = \mathbf{r}_0$.

**Theorem 2.** *If all players adopts LBB$(\alpha, \beta)$ bitrate adaptation policy, the following statements are true:*

1. $(\mathbf{r}_0, \mathbf{w}_0): \; r_{i0} = w_{i0} = \frac{w}{n}$ *is an equilibrium;*
2. *If* $-\frac{1}{n} < \frac{\partial h_i(\mathbf{r}_0)}{\partial r_j} < 0$, $\forall i \neq j$, *then the equilibrium is locally asymptotically stable;*
3. *If* $\frac{\partial h_i(\mathbf{r}_0)}{\partial r_j} < -\frac{1}{n}$, $\forall i \neq j$, *then the equilibrium is unstable;*

*Proof.* Combining buffer-based bitrate controller with buffer dynamics, we get:

$$r_i[k+1] = r_i[k] - \alpha \Delta T + \frac{w_i[k] \alpha \Delta T}{r_i[k]}, \quad \forall i \in \mathcal{P} \tag{2.34}$$

We linearize the system about the equilibrium and let $\delta r_i = r_i - r_{i0}$.

$$\begin{aligned}\delta r_i[k+1] &= \delta r_i[k]\left(1 + \frac{\alpha\Delta T}{r_{i0}^2}\left(\frac{\partial h_i(\mathbf{r}_0)}{\partial r_i} r_{i0} - h_i(\mathbf{r}_0)\right)\right) \\ &\quad + \sum_{j\neq i} \delta r_j[k]\left(\frac{\alpha\Delta T}{r_{i0}} \cdot \frac{\partial h_i(\mathbf{r}_0)}{\partial r_j}\right) \\ &= \delta r_i[k]\left(1 + \frac{\alpha\Delta T}{r_{i0}}\left(\frac{\partial h_i(\mathbf{r}_0)}{\partial r_i} - 1\right)\right) \\ &\quad + \sum_{j\neq i} \delta r_j[k]\left(\frac{\alpha\Delta T}{r_{i0}} \cdot \frac{\partial h_i(\mathbf{r}_0)}{\partial r_j}\right)\end{aligned}$$

Note that as $\sum_{i\in\mathcal{P}} w_i = w$, we have

$$\sum_{j\in\mathcal{P}} \frac{\partial h_i(\mathbf{r})}{\partial r_j} = 0$$

If we let $\frac{\partial h_i(\mathbf{r}_0)}{\partial r_j} = c, \forall j \neq i$, we have $\frac{\partial h_i(\mathbf{r}_0)}{\partial r_i} = -(n-1)c$. Combining with the linearized system equation, we get the following equation in matrix form:

$$\delta\mathbf{r}[k+1] = \mathbf{A} \cdot \delta\mathbf{r}[k]$$

where

$$\mathbf{A} = \left(1 + \frac{\alpha\Delta T}{r_{i0}}(-nc-1)\right) \cdot \mathbf{I} + \left(\frac{\alpha\Delta T}{r_{i0}} \cdot c\right) \mathbf{1}\mathbf{1}^T$$

The eigenvalues of matrix $\mathbf{A}$ are:

$$\sigma(A) = \left\{1 + \frac{\alpha\Delta T}{r_{i0}}(-nc-1), 1 - \frac{\alpha\Delta T}{r_{i0}}\right\}$$

From the second eigenvalue we have a constraint that couples $\alpha, \Delta T$ and $r_{i0}$

$$\frac{2r_{i0}}{\alpha\Delta T} > 1$$

Using indirect Lyapunov method, if $c < -1/n$, the linearized system is unstable about the equilibrium, then the original non-linear system is unstable. On the other hand, if $-1/n < c < 0$, the equilibrium is locally asymptotically stable. $\square$

Derived results for the fairness of homogeneous video players (both RB and BB) yield intuitive steady state solution. In a multiplayer system composed of N identical video players one can expect for the optimal steady state policy be sharing the total bandwidth equally between all the video players.

Note that comparing results in homogeneous RB and BB players, we found that the convergence of RB players depends on both bandwidth allocation and bitrate adaptation policies, while convergence of BB players only depends on bandwidth allocation functions. The key reason is that, the bitrate decisions of BB players reflects the state of the player, i.e., buffer level, while the bitrate decisions of RB players does not depend on the internal state of the buffer.

**Fairness of heterogeneous BB players:** We consider the case where all players are adopting different buffer-based bitrate adaptation policies but have the same QoE function.

**Theorem 3.** *If player $i$ adopts $LBB(\alpha_i, \beta), i \in \mathcal{P}$ bitrate adaptation policy, the following statements are true:*

1. $(\mathbf{r}_0, \mathbf{w}_0): \; r_{i0} = w_{i0} = \frac{w}{n}$ *is an equilibrium;*

2. *If* $\frac{1}{(\sum_i \alpha_i)/(\max \alpha_i) - n} < \frac{\partial h_i(\mathbf{r}_0)}{\partial r_j} < 0$, $\forall i \neq j$, *then the equilibrium is locally asymptotically stable;*

3. *If* $\frac{\partial h_i(\mathbf{r}_0)}{\partial r_j} < \frac{1}{(\sum_i \alpha_i)/(\max \alpha_i) - n}$, $\forall i \neq j$, *then the equilibrium is unstable;*

*Proof.* Similarly to the proof of Theorem 2 by combining buffer-based bitrate controller with buffer dynamics we get the following expression:

$$r_i[k+1] = r_i[k] - \alpha_i \Delta T + \frac{w_i[k]\alpha_i \Delta T}{r_i[k]}, \quad \forall i \in \mathcal{P} \tag{2.35}$$

We linearize the system about the equilibrium and let $\delta r_i = r_i - r_{i0}$.

$$\begin{aligned} \delta r_i[k+1] = \delta r_i[k] \left(1 + \frac{\alpha_i \Delta T}{r_{i0}} \left(\frac{\partial h_i(\mathbf{r}_0)}{\partial r_i} - 1\right)\right) \\ + \sum_{j \neq i} \delta r_j[k] \left(\frac{\alpha_i \Delta T}{r_{i0}} \cdot \frac{\partial h_i(\mathbf{r}_0)}{\partial r_j}\right) \end{aligned}$$

Given $\sum_{i \in \mathcal{P}} w_i = w$, we have $\sum_{j \in \mathcal{P}} \frac{\partial h_i(\mathbf{r})}{\partial r_j} = 0$. If we let $\frac{\partial h_i(\mathbf{r}_0)}{\partial r_j} = c$, $\forall j \neq i$, we have $\frac{\partial h_i(\mathbf{r}_0)}{\partial r_i} = -(n-1)c$. As in Theorem 2, combining the previous expression with the linearized system equation, we get the same matrix equation $\delta \mathbf{r}[k+1] = \mathbf{A} \cdot \delta \mathbf{r}[k]$ where

$$\mathbf{A} = \left(1 + \frac{\alpha_i \Delta T}{r_{i0}}(-nc-1)\right) \cdot \mathbf{I} + \left(\frac{\alpha_i \Delta T}{r_{i0}} \cdot c\right) \mathbf{1}\mathbf{1}^T$$

The parameter $\alpha_i$ is different in every rowi, $\in \mathcal{P}$.The eigenvalues of the first part of matrix $\mathbf{A}$ are:

$$\sigma(A_1) = \left\{1 + \frac{\alpha_i \Delta T}{r_{i0}}(-nc-1)\right\}, \forall i \in \mathcal{P}$$

The eigenvalues of the second part of matrix $\mathbf{A}$ are

$$\sigma(A_2) = \left\{\underbrace{0, 0, \cdots 0}_{n-1}, \frac{c\Delta T}{r_{i0}} \sum \alpha_i \right\}$$

In this case the eigenvalues of $\mathbf{A}$ depend on $\alpha_i > 0, \forall i \in \mathcal{P}$. Lets assume that $\max(\alpha_i) = \alpha_k$ and $\min(\alpha_i) = \alpha_l$. Now, we can lower and upper bound the $\sigma(A)$. The lower bound is

$$\sigma_{min}(A) > 1 + \frac{\alpha_l \Delta T}{r_{i0}}(-nc - 1)$$

and upper bound is

$$\sigma_{max}(A) < 1 + \frac{\alpha_k \Delta T}{r_{i0}}(-nc - 1) + \frac{c \Delta T}{r_{i0}} \sum \alpha_i.$$

Now, using the indirect Lyapunov method we obtain the stability boundaries. The given system is stable if both the lower and the upper bound is within the unit circle. From here we get $c \in \left( \frac{1}{\sum \frac{\alpha_i}{\alpha_k} - n}, 0 \right)$.

$\square$

**Implications on system design:** From the analysis we know that, in homogeneous player case, the convergence of BB players only depends on the characteristics of the bandwidth allocation function $h(\cdot)$, while for RB players, the convergence depends on the composite of bandwidth allocation function $h(\cdot)$ and player adaptation algorithms $f(\cdot)$. This has the following key implications that informs the system design:

First, the analysis confirms that the router-side bandwidth allocation function is critical to the convergence of both RB and BB players. Given that the player adaptation algorithms are designed by potentially different providers and may not be considering multiplayer effect, it could in turn be beneficial to redesign the bandwidth allocation function to ensure convergence with a larger range of player adaptation algorithms.

Second, the analysis provides a theoretical guide for the design of RB player adaptation algorithms which helps us better understand why existing design (such as [47]) works. Given that the convergence depends on both bandwidth allocation and player adaptation, if TCP-based implicit bandwidth allocation is hard to change, we can design better player adaptation algorithms so that $h \circ f$ is contractive. One example of this principle is the design of FESTIVE [47], where $f(\cdot)$ function is concave to make sure $h \circ f$ is contractive.

### 2.4.3 NMPC-based Router-Assisted Bandwidth Allocation for QoE Fairness

Despite a fully distributed scheme, the analysis from the previous section has posed the fundamental limitation of TCP-based bandwidth allocation scheme: First, not all $h(\cdot)$ lead to convergence to QoE fairness in steady state even if players have the same QoE function $U(\cdot)$ and use the same class of bitrate adaptation policies $f(\cdot)$. Second, it cannot take into account different QoE goals and will not converge to fairness when players employ different classes of bitrate adaptation policies. As such, in order to achieve multiplayer QoE fairness, we want to design better player bitrate adaptation policies $f_i(\cdot)$ and bandwidth allocation policy $h(\cdot)$.

However, it is difficult to deploy/modify bitrate adaptation policies of all video players as they belongs to different and competing video streaming services, e.g., Netflix, YouTube, Amazon Video, etc. Also, controlling the bandwidth from the player side is difficult as the player runs on top of HTTP and cannot change the underlying TCP protocol. Instead, routers are in a good position to collect information of each player and video stream, and can technically control the bandwidth allocation. As smart routers are becoming more and more pervasive in the home entertainment industry (e.g. Google OnHub router), we envision that router-assisted bandwidth allocation scheme is more practical. Overall, we develop a hybrid router-assisted control for fairness: we keep the player adaptation policies $f_i(\cdot)$ unchanged, and design bandwidth allocation policy $h(\cdot)$ to achieve QoE fairness.

As routers have access to all video streams going through, we assume it can get or learn the following information from each player $i \in \mathcal{P}$: 1) current states of the player including bitrate $r_i$, buffer level $b_i$, 2) bitrate adaptation policy $f_i(\cdot)$, 3) QoE function $U_i(\cdot)$.

Given these information, the router-side bandwidth allocation function $h(\cdot)$ is obtained implicitly by solving the following bandwidth allocation problem in a moving horizon manner, regarding each player as a closed-loop system.

$$\begin{aligned} \max \quad & F\left(U_1, \cdots, U_N\right) \\ \text{over} \quad & \mathbf{w}[k] \quad \text{given } W[k], k \in \mathcal{K} \\ s.t. \quad & (2.24) - (2.27) \\ & r_i[k] = f_i(w_i[k-1], b_i[k]), \quad \forall i \in \mathcal{P}, k \in \mathcal{K} \end{aligned}$$

Note that as the dynamics of players are non-linear, the resulting controller is a non-linear MPC-based controller.

### 2.4.4 Evaluation

#### Evaluation Setup

**Evaluation framework:** We employ a custom Matlab-based simulation framework. The duration of each time step is 2s and the simulation framework works in a synchronized manner: At the beginning of each 2s interval, the states of the player and the network is updated according to player dynamics and previously recorded traces. The bitrate and bandwidth decisions are then made simultaneously. There is no event in between each 2s interval. Note that this is slightly different from the single-player simulation in previous section as the player decisions are not synchronized, i.e., the player can change the bitrate at chunk boundaries, which may not necessarily be every 2s. We acknowledge this limitation and will test in real asynchronized settings in future work.

**Resource allocation schemes:** We compare the following algorithms:

1. *Baseline*: In baseline scheme, the bandwidth controller knows the $q(\cdot)$ function of all players, and the bandwidth is allocated by solving the steady-state bandwidth allocation problem at the beginning of each time step. Given allocated bandwidth, each player then adopts RB or BB adaptation strategies to choose its bitrate. This scheme has been seen in recent work [31, 36, 58].

2. *Router*: In router-assisted scheme, the bandwidth controller knows the QoE functions, states (buffer level, bitrate), and bitrate adaptation strategies of all players. The router-assisted bandwidth controller works in a moving horizon way: At the beginning of each time steps, the controller predict the bandwidth in a fixed horizon to the future, and solve the router-assisted bandwidth allocation problem (BWA) in the horizon to decide bandwidth allocation. In this work, we assume the bandwidth is given and the bandwidth allocation is calculated in one shot instead in a moving horizon way.

3. *Centralized*: The centralized scheme entails calculating the optimal bandwidth allocation and the bitrate decisions simultaneously by solving the joint optimization problem. We assume the controller knows the entire future bandwidth. While less practical, the centralized controller provides us with an upper bound of the performance.

**Metrics:** We evaluate the algorithms using the following performance metrics:

1. *$\alpha$-fairness*: We adopt $\alpha$-fairness measure as it is widely used in prior work [53]. Specifically, we focus on two special case of $\alpha$-fairness: 1) $\alpha = 0$ corresponding to *social welfare, sum of QoE, or efficiency*; 2) $\alpha = 1$ corresponding to *proportional fairness.* As $\alpha$-fairness can be decomposed into a component corresponding to efficiency and

another component corresponding to fairness measures that does not depend on fairness [53], we also use social welfare and normalized Jain's index as detailed metrics.

2. *Social welfare*: Defined as sum of QoE of all players, i.e., $\sum_{i\in\mathcal{P}} U_i$.

3. *Normalized Jain's index*: Defined as the Jain's index [45] of normalized QoE, namely, Jain's index of $\mathbf{U}/(\sum_{i\in\mathcal{P}} U_i)$. Jain's index is widely used in prior work to depict QoE fairness of players [24, 47, 53], it is defined as $J(\mathbf{x}) = (\sum x_i)^2/(n \cdot \sum x_i^2)$. Note that instead of Jain's index, we could choose any other fairness measure that satisfies all axioms as shown in [53]. However, here we show the fairness result in terms of Jain's index just in order to be consistent with prior work [24, 47].

**Throughput traces:** We use the throughput trace from FCC MBA 2014 project [9]. The dataset has more than 1 million sessions of throughput measurement, each containing 6 measurement of 5-sec average throughput. For experiment purposes, we concatenate the measurements from the same client IP and server IP, and use the concatenated traces in the experiment. To avoid trivial cases where the available bandwidth is too high or too low, we only use traces whose average throughput is 0 to 3Mbps. Also, we multiply the throughput by the number of players in the experiment to eliminate the scaling effect in multiplayer experiments.

**Player pararmeters:** The time horizon is discretized by $\Delta t = 2s$. For simplicity, we assume players can choose bitrate in a continuous range [200kbps, 3000kbps]. We set buffer size to be 30s. For QoE functions, we set $\mu = 1$ for all players. For default settings, players has the following video quality function $q(r) = r^p$, we set $p = 0.6$ by default, making $q(\cdot)$ function concave. Note that this can be non-concave in general, e.g., we could also use the sigmoid-like functions as suggested in [29], however, this will make the objective non-convex. We let RB players adopt $r[k] = 0.8 \times w[k-1]$, while BB players adopt $r[k] = 100 \times b[k]$ by default.

## End-to-End Results

In this section, we focus on the end-to-end comparison of the algorithms.

**Efficiency-vs-fairness tradeoff:** We first evaluate the algorithms in terms of normalized social welfare (sum of QoE) and normalized fairness measure (Jain's index). We change $\alpha$ in $\alpha$-fairness in order to get different points on the curve. Figure 2.15 shows the pareto front of the algorithms. There are three observations: First, router-assisted control outperforms baseline controller by 5-7% in terms of social welfare given the same normalized Jain's index. For example, if we let normalized Jain's index to be 0.8, router assisted controller achieves 56% of optimal, while baseline controller only achieves 50% of optimal. Second, centralized controller significantly outperforms both router-assisted and baseline controller with 15+% advantage. This is because centralized controller has more flexibility on deciding the bitrate for each player, while router-assisted controller does not have direct control over players' bitrates and can only steer the bitrate by controlling the bandwidth (for RB players) and implicitly buffer level (for BB players). Third, we observe a natural tradeoff between social welfare and fairness. According to Lan et al. [53], $\alpha$-fairness can be factored into two component: efficiency (social welfare) and fairness measure that satisfies the five axioms and does not depend on scale. When $\alpha = 0$, both centralized and router-assisted controller optimizes social welfare without considering the fairness of players. As such, the social welfare at the left most point of the curve is at the maximum. However, as $\alpha$ is increased, more and more weight is put on the fairness of QoE, leading to increased fairness but less total QoE. Note that this resonates with the observation in prior work [47] on the tradeoff between sum of bitrates and their fairness, but our proposed algorithms are able to systematically adjust this tradeoff by selecting an appropriate $\alpha$.

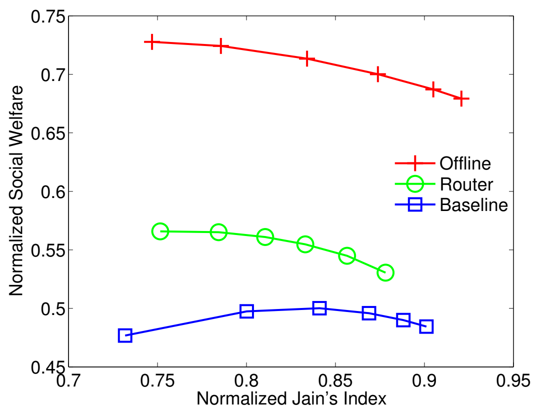
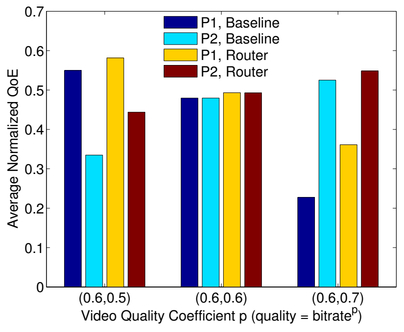


Figure 2.15: Social welfare vs fairness tradeoff

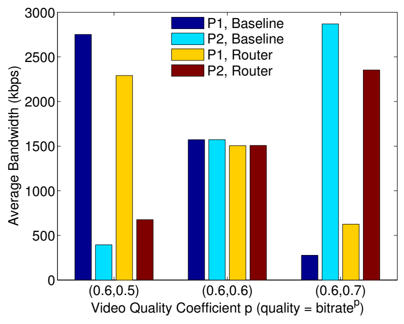
### Sensitivity Analysis

Next, we conduct sensitivity analysis with respect to key parameters so as to understand the robustness and the reason why router-assisted controller outperforms existing methods.

**Impact of QoE functions:** We first look at how the algorithms performs under different QoE functions in Figure 2.16a. We use two BB players with the same parameters except for video quality functions, i.e., $q(\cdot)$ function. We let $q(r) = r^p$ and vary the coefficient $p$. The larger $p$ is, the user-perceived quality is more sensitive w.r.t. bitrate; The smaller $p$ is, the less sensitive the user is to bitrate. As shown in Figure 2.16b, both baseline and router assisted controller allocate more bandwidth to the player with larger $p$ and thus requiring higher bitrate, as both controllers takes into account the $q(\cdot)$ function in their optimization. However, router-assisted algorithm outperforms baseline controllers as it considers player buffer dynamics and lead to faster convergence to optimal bitrates. In addition, the advantage of router-assisted algorithm over baseline controller is increasing as the video quality coefficients $p$ for different players become more diverse. Note that this confirms our observation that more bandwidth should be allocated to high-resolution devices in order to achieve QoE fairness.



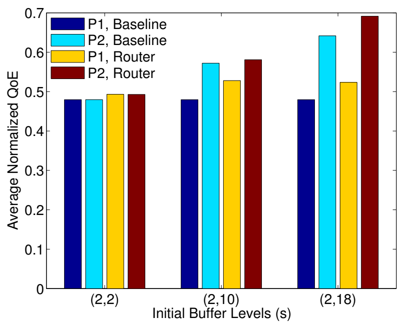
(a)



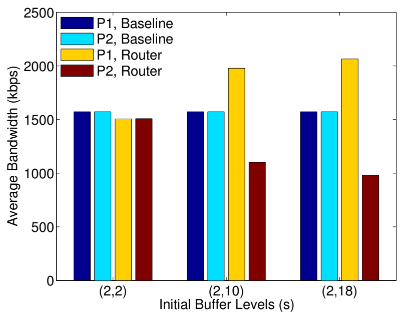
(b)

Figure 2.16: Impact of QoE functions

**Impact of initial conditions:** We further investigate how the players' initial buffer levels impact the performance. Figure 2.17a shows the players' normalized QoE vs different initial conditions, while Figure 2.17b shows the bandwidth allocated to players in baseline and router-assisted schemes. There are three key observations: First, the router-assisted algorithm consistently outperforms baseline solution, increasing the normalized QoE for each player. Second, the router-assisted algorithm has more advantage over baseline solution when the initial buffer levels for the players become more diversed. For instance, while router-assisted and baseline achieves similar performance when both players have 2s buffer initially, both players' QoE are significantly improved when initial buffer levels are 2s and 18s respectively. Third, an interesting observation from Figure 2.17b is that, while baseline solution does not consider states and dynamics of the players and therefore allocate the same bandwidth to both players even one player has much more buffer and need less bandwidth, router assisted algorithm allocate less bandwidth to players with full buffer and more bandwidth to player with empty buffer as it needs to quickly accumulate buffer so as to stream at high bitrate. As such, router-assisted algorithm achieves better performance as it takes into account the states and dynamics of the players, which is critical to players' QoE.



(a)



(b)

Figure 2.17: Impact of initial conditions

### Summary of Results

Our main findings are summarized as follows:

1. Given fixed normalized Jain's index, router-assisted algorithm ourperforms baseline solution by 5-7% in terms of social welfare (sum of QoE), while centralized bandwidth allocation + bitrate control achieves 70% of optimal, achieving 15+% advantage comparing to other solutions.

2. Our sensitivity analysis shows that router-assisted algorithm has more advantage over baseline solution when the QoE functions and initial conditions of players are more diverse. Moreover, router-assisted algorithm can allocate more bandwidth to players with less buffer while baseline solution fails to take into account the states of the players.

## 2.5 Summary

Our work was motivated by recent debates surrounding the design of dynamic adaptive streaming over HTTP (DASH) algorithms. To bring some rigor to this space, we developed a control-theoretic problem formulation that allowed us to explore the design space systematically and evaluate quantitatively different classes of solutions through well-defined QoE metrics. With the key insights that a broader design space is available compared to existing solutions, we designed and implemented a model predictive control approach to optimally combine buffer occupancy and throughput predictions in order to maximize the user's QoE. We demonstrated a practical implementation of MPC using the `dash.js` reference video player. Our trace-driven emulations using realistic throughput variability traces confirmed the advantages over state of the art solutions in a wide range of operating conditions with negligible increase in computation and memory requirements.

Instead of regarding available bandwidth as given by a black box, we further consider the multiplayer interaction in adaptive video streaming, namely, the joint bandwidth allocation and bitrate adaptation problem with single bottleneck. We build a mathematical model and conduct theoretical analysis on the convergence of RB/BB players under unideal TCP assumptions. Given that convergence is not guaranteed in general, we develop a router-assisted control which allocate bandwidth to players taking into account their bitrate adaptation strategies and states. Using trace-drive simulations, we show that our proposed router-assisted control outperforms existing QoE-aware bandwidth allocation algorithms as it can adaptively allocate bandwidth to players with high resolution and in more urgent need to accumulate buffer.

# Chapter 3

# Case Study in Electric Power Networks

## 3.1 Overview

Today's electrical power system has seen an increasing presence of uncertaintes in its daily operations, such as equipment failures, demand variations, and unpredictable power output of renewable resources. These uncertainties can lead to inefficiency and potential risks in power system operations if not well managed [44]. Various approaches have been proposed by the researchers to manage the uncertainties and related risks at the system level [26, 44].

Electric power generators have diverse dynamic characteristics and control capabilities which impact their ablilities to adjust power output to desired value in power system operations. Depending on the responsiveness and accuracy to system operators' commands, we roughly divide them to slow and fast generators. In addition to the well-studied variability of demand and renewable generation resources, slow generator's deviation from the system operator's schedule has been recently identified as a new type of uncertainty which may lead to risks the real-time operations of the grid [44].

Specifically, in today's power system, online power balancing is achieved by combined efforts of real-time dispatch, which is conducted every 5 minutes before the actual interval, and automatic generation control (AGC), during the 5-min slot. While slow generators can commit to certain amount of generation in real-time dispatch stage, their actual generation during the actual time interval may deviate from the agreed schedule due to physical limitations and gaming.

These deviations lead to new risks in online power balancing: From the (independent) system operators' ((I)SO) perspective, these deviations create non-zero mean imbalances in real-time, which needs more expensive AGC resources to compensate; From market participants' point of view, neither the generators with large uncertain deviations get penalized under the current real-time market rule, nor the generators which follows the schedule better get rewarded. As a result, if this risk is not well managed, it not only adds stress to the AGC system, but also leads to inefficiency and unfairness in the real-time energy market and ancillary services market. In this light, how to manage the risks created by slow generator's deviation becomes a fundamental question posed for the (I)SOs.

The risks can be managed by the (I)SO in a centralized way, or by the market participants, i.e., the generators, in a distributed, market-based way. A centralized framework has recently been proposed in [44] to systematically manage the risks created by slow generator's deviations in online power balancing. Instead of the currently adopted real-time dispatch + AGC 2-stage approach, [44] proposes a novel 3-stage framework comprising: 1) risk-based real-time dispatch before the 5-min interval, 2) redispatch of fast generators at the beginning of the actual interval, and 3) AGC during the interval. The characteristics of slow generators' deviation are first statistically learned from historical data and directly by used the (I)SO to dispatch slow generators taking into account their potential deviations. Right at the beginning of the 5-min interval, actual deviations are observed and then compensated by re-dispatching fast-responsive generators. The imbalances left are then covered by AGC.

While the centralized framework hides the complexity of risk management from the market participants, it is incomplete in the sense that it has not defined a transparent pricing and cost allocation mechanism. To this end, in this dissertation, we design a novel market structure to manage the risks.

We develop a risk-based market rule in which the slow generators with deviations are charged with the corresponding redispatch cost due to their deviations. Under this market rule, the slow generators consider and internalize the risks of potential deviations when they create their “simple bid” functions in real-time balancing market. By conducting conventional deterministic real-time dispatch based on “simple bids”, the (I)SO can reduce the risks created by deviations. In other words, the risks of deviations are managed distributedly by the market participants when they create the bid function, instead of being considered by (I)SO in a centralized way in risk-based real-time dispatch.

We conduct numerical simulations to show that the proposed risk-based market structure 1) improves market efficiency by penalizing the generators with large deviations while rewarding generators with less uncertainties according to market-based prices; 2) reduces total generation cost as well as the randomness of the cost.

## 3.2 Background

In this section, we first provide a brief survery of risks in current electricity market, and then identify new risks brought by slow generators' deviations. We introduce related work in electricity market formulations and argue that a new risk-based formulation is necessary to address the new risks.

### 3.2.1 Risks in Today's Electricity Markets

There are several types of risks that exist in today's electricity market. Here we give a brief survey of the risks.

1. *Equipment failure* [26]. This includes generator failure and disconnection of transmission lines. For example, operating generators may experience unexpected failure and are not able to provide power output. Transmission lines may fail to work due to overloading. These failures may pose significant risks in the electricity market as they may substantially change the total power supply and the power flow in transmission networks. If not well managed, such failures can lead to cascading failures, e.g., the failure of one transmission line can result in the routing of power flow to other lines, leading to potential overloading and new line failures. The issue of the equipment failure is also known as *reliability* issue. Note that we are not considering reliability-related risks in this thesis.

2. *Uncertainty of renewable resources* [72, 76]. Renewable resources such as wind and solar generators are known to be highly volatile and unpredictable. While in the dispatching stage the renewable resources are considered according to predicted generations, the actual generation may differ from the schedule and result in significant supply-demand imbalance.

3. *Uncertainty of demand* [48, 49]. The amount of power demand from users is also volatile and hard to predict. For example, a public sport event may significantly increase the power demand for a short period of time, which will need more generations.

While these risks have been well studied by prior work [26, 48, 49, 72, 76], in this thesis, we study a new type of risks posed by slow generators' deviations.

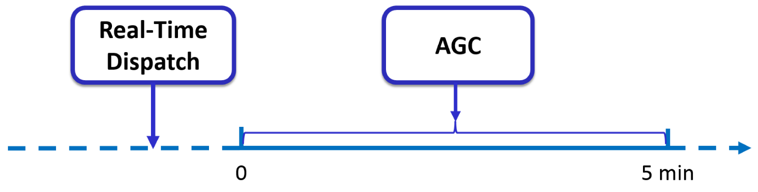


Figure 3.1: Today's online power balancing scheme

| Generator | Cost | Mean of $\Delta P_i$ | Std of $\Delta P_i$ |
|---|---|---|---|
| 1 | $50P_1^2 + 80P_1$ | $-20\%$ | $10\%$ |
| 2 | $100P_2^2 + 88P_2$ | $-15\%$ | $5\%$ |
| 3 | $200P_3^2 + 100P_3$ | 0 | $5\%$ |
| 4 | $600P_4^2 + 258P_4$ | 0 | 0 |
| 5 | $600P_5^2 + 300P_5$ | 0 | 0 |

Table 3.1: Generator profiles: 1-3 are slow generators, 4,5 are fast generators

### 3.2.2 Risks Posed by Slow Generators' Deviations

We start by identifying the new risk in today's online power balancing scheme. Figure 3.1 shows the current online power balancing scheme: Real-time dispatch (RTD) is carried out every 5 minutes before the actual interval, while AGC is then used during the 5-min interval to cover the imbalances left in real-time.

However, the deviation of generators from schedule can lead to potential risks in today's online power balancing scheme: While the generators are scheduled for certain amount of generation in real-time dispatch stage, the actual generation during the 5-min interval can deviate from the schedule, because of physical limitations and gaming (strategically not following the command for its own profit). As a result, the (I)SO needs to utilize more AGC to cover the imbalances caused by deviations, leading to increased stress in AGC system. In addition, while the increased need of AGC is caused by deviations of generators, these generators are not penalized for not following the schedule under the current settlement process of real-time energy market or ancillary service market, which leads to increased market inefficiency and unfairness.

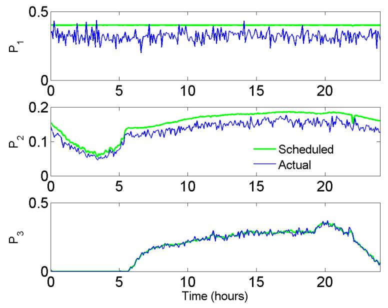
We use an example to illustrate the problem in more detail. In the test system, generator 1-3 can deviate from the (I)SO's RTD command, and are thus called *slow generators*. On the other hand, generator 4-5 are called *fast generators*, since they can exactly follow the (I)SO's command and adjust their generation output very quickly. Figure 3.2a shows the scheduled and actual generation of slow generators 1-3, respectively. As a result, the (I)SO sees the total non-zero mean imbalances shown in Figure 3.2b, which must be compensated by the more expensive AGC resources in real-time.

The identified issue with today's online power balancing then poses the following fundamental questions to the (I)SOs:

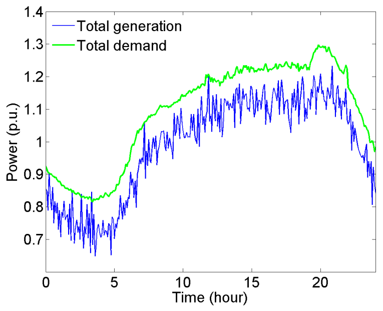
1. How to reduce the generators' deviations and the resulting increasing need for expensive AGC resources?
2. How to design market rules to allocate the cost of online power balancing to reward the right technologies for what they do?

While previously proposed centralized risk-based real-time dispatch [44] provides an initial answer to these questions, no market rule has been designed to facilitate fair allocation of cost of risks according to the causation. In this dissertation, we develop a risk-based market structure to answer these questions. The desired risk-based market rule should:

1. Penalize generators with large deviations and reward those which follow the schedule better;
2. Allow different generators to have different risk preferences;
3. Drive slow and fast generators to the optimal operating conditions such that total generation cost is minimized.



(a) Slow generators' deviations from the schedule



(b) Total imbalance seen by (I)SO

Figure 3.2: Supply-demand imbalance as a result of individual generator's deviation from schedule

In this thesis, we consider generators to be the only market participants while demand is given/known. An interesting future work will be extending the proposed framework to include demand resources as market participants.

### 3.2.3 Related Work in Electricity Market Formulations

There has been a lot of existing work on the mathematical foundations of the electricity market [32, 49, 62, 63]. The determination of generation schedules and settlement process can be formulated as an optimal power flow problem minimizing the total generation cost, also known as economic dispatch problem. Using the lagrangian multipliers in the problem, the (I)SO can calculate the locational marginal prices and decide payments. Specifically, Joo et al. [49] proposed the use of principles of decomposition in deterministic economic dispatch.

Risks have been considered in electricity market formulations by extending the deterministic optimization into stochastic optimization. For example, Joo et al. [48] considers the risks of uncertainty of demand, while Zhang et al. [76] proposed risk-limiting dispatch to consider the uncertainty of renewable resources.

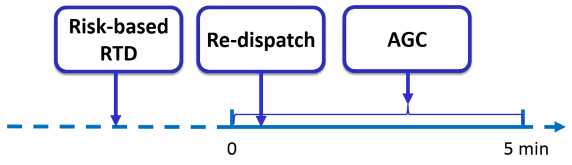


Figure 3.3: The new risk-based online power balancing scheme

However, the risks of slow generators' deviations have not been studied in the electricity market formulations. Next, based on the existing work on deterministic optimization and risk-based optimization, we develop a risk-based dispatch that considers the risks of slow generators' deviations.

## 3.3 Centralized Risk-Based Real-Time Dispatch

In this section, we propose a centralized risk-based RTD approach as an extension to conventional RTD to consider generators' possible deviations.

Figure 3.3 provides an overview of the proposed risk-based online power balancing scheme. Instead of the conventional 2-stage approach, the risk-based method consists of 3 stages: Risk-based real-time dispatch, re-dispatch, and AGC. Before the 5-min interval, the (I)SO first predicts possible deviations of slow generators based on historical data, and dispatch slow generators taking into consideration the potential costs due to the deviations. At the beginning of the actual interval, the (I)SO measures the actual deviations of slow generators and adjusts fast generators to compensate the deviations. The imbalance left is then compensated by AGC during the 5-min interval. In the rest of the section, we introduce each of the stages in more detail.

**Modeling slow and fast generators:** First, we develop a formal model characterizing slow and fast generators. We model slow generators as generators that can have deviation of power output $\Delta P_i$ in the actual time interval, given the scheduled output $P_i$. While in practice the deviation can be the result of generators physical limitation (e.g., poor govenor response) or strategic behavior (gaming), in this thesis we only consider physical limitations and generators are not able to control their deviations in the actual interval and can only regard it as random. We assume the deviation follows a Gaussian distribution: $\Delta P_i / P_i \sim \mathcal{N}(\mu_i, \sigma_i)$. Ideally, real market data is needed to verify the assumption of distribution of deviations and learn the parameters $(\mu_i, \sigma_i)$. However, we acknowledge that our model is based on conversation with industry personnel and not based on real data. Instead of building data-driven model, our goal in this thesis is to adopt a model to provide a proof-of-concept analysis of the impact of the proposed two-stage dispatch. We regard real-data-based analysis as future work elaborated in detail in Chapter 5. We assume each slow generators to be owned by different entities. We model the fast generators as generators that can quickly respond to the (I)SO's command at the beginning of the actual interval. The fast generators does not have deviations and only participate the re-dispatch stage as illustrated later in the thesis.

**Risk-based real-time dispatch:** Different from conventional RTD, the potential risks resulting from slow generator's deviations are considered in the centralized risk-based RTD approach. Given this statistical information of the deviations, the centralized risk-based real-time dispatch problem is then formulated as follows, where the optimal generation of slow generators, $P_{i,t}, i \in \mathcal{S}$ is found to minimize the conditional value-at-risk of the generation cost of slow generators as well as potential redispatch cost of fast generators:

$$\min_{P_{i,t}, i\in\mathcal{S}} \quad CVaR_\beta \left( \sum_{i\in\mathcal{S}} C_{S,i}(P_{i,t} - \Delta P_{i,t}) + R(\Delta P_t) \right) \tag{3.1}$$

$$s.t. \quad \sum\nolimits_{i\in\mathcal{S}} P_{i,t} = P_D \tag{3.2}$$

$$|F| \le \overline{F} \tag{3.3}$$

$$|P_{i,t} - P_{i,t-1}| \le RR_i \tag{3.4}$$

$$\underline{P}_i \le P_{i,t} \le \overline{P}_i \tag{3.5}$$

$$\Delta P_{i,t} | P_{i,t} \sim f(\Delta P_{i,t}) \tag{3.6}$$

where $C_{S,i}(\cdot)$ is the generation cost function of slow generator $i$, $R(\Delta P_t)$ is the redispatch cost given *actual* deviations of slow generators $\Delta P_t$.

Since the actual deviation $\Delta P_t$ is unknown before the actual 5-min interval begins, the actual generation cost $C_{S,i}(P_{i,t} - \Delta P_{i,t})$ and the redispatch cost $R(\Delta P_t)$ are both stochastic at this point. We use the concept of *conditional value-at-risk (CVaR)* to quantify the cost, defined as

$$CVaR_\beta(cost) = \mathbb{E}\left(cost - VaR_\beta(cost)\right)_+ \tag{3.7}$$

where $VaR_\beta(cost) = \min\{L | Prob(cost \le L) \ge \beta\}$ is the value-at-risk of the cost. CVaR is a widely-adopted coherent risk measure [66]. $\beta \in [0, 1]$ repensents the risk preference of the (I)SO. The more $\beta$ is, the more risk is perceived which leads to more conservative decisions. In extreme cases, if $\beta = 0$, the CVaR degenerates to the average cost; On the other hand, if $\beta = 100\%$, the CVaR becomes the worst-case cost.

**Redispatch:** In the redispatch stage, the actual deviations of slow generators $\Delta P_{i,t}$ are measured and compensated by redispatching fast generators. The corresponding redispatch cost $R(\Delta P_t)$, as well as the generation of fast generators, are calculated as follows:

$$R(\Delta P_t) = \min_{P_{j,t}, j \in \mathcal{F}} \quad \sum\nolimits_{j \in \mathcal{F}} C_{F,j}(P_{j,t}) \tag{3.8}$$

$$s.t. \quad \sum\nolimits_{j \in \mathcal{F}} P_{j,t} = \sum\nolimits_{i \in \mathcal{S}} \Delta P_{i,t} \tag{3.9}$$

$$|F| \leq \overline{F} \tag{3.10}$$

$$|P_{j,t} - P_{j,t-1}| \leq RR_j \tag{3.11}$$

$$\underline{P}_j \leq P_{j,t} \leq \overline{P}_j \tag{3.12}$$

While the centralized risk-based dispatch provides a framework for risk management in online power balancing, further work is still needed: First, the centralized approach acts as a blackbox and does not have a transparent corresponding cost allocation or pricing mechanism. Second, in the centralized scheme, the (I)SO needs to learn the possible deviations of slow generators from historical data, which can be inaccurate and lead to sub-optimal decisions. Third, in real-world scenarios, different generators can have different risk preferences, which is difficult to incorporate in the centralized approach. To this end, a risk-based market structure is needed to bring the risk-based dispatch to practice.

## 3.4 Risk-Based Market Design

In this section we develop a novel risk-based market structure to manage the risk caused by slow generator's deviations. We first explain the new risk-based market settlement process. After that, we elaborate how the simple bids of slow generators are created by formulating a profit maximization problem. We compare the market-based risk management and centralized approach at the end of the section.

| Revenue | Slow generator $i$ | Fast generator $j$ | Load |
|---|---|---|---|
| RTD stage | $\lambda P_i$ | 0 | $-\lambda P_D$ |
| Redispatch stage | $-\alpha \Delta P_i$ | $\alpha P_j$ | 0 |
| Total revenue | $\lambda P_i - \alpha \Delta P_i$ | $\alpha P_j$ | $-\lambda P_D$ |

Table 3.2: Risk-based settlement process: $\lambda$ is the RTD price while $\alpha$ is the redispatch price

### 3.4.1 Market Structure

In order to equip risk-based online power balancing scheme shown in Figure 3.3 with a market structure, we propose the following risk-based market rule:

**Risk-based market rule:** Slow generator $i$ are paid $\lambda P_i$ according to RTD schedule $P_i$ and price $\lambda$, but are charged with $\alpha \Delta P_i$ based on its actual deviation $\Delta P_i$ and price $\alpha$ in the redispatch stage at the beginning of the 5-min interval. Its total revenue is then $\lambda P_i - \alpha \Delta P_i$. Fast generator $j$ are paid $\alpha P_j$ where $P_j$ is its generation in redispatch stage. Demand pays $\lambda P_D$.

The risk-based market rule is summarized in Table 3.2. By allocating the redispatch cost to the deviating slow generators, the market structure creates economic incentives for the slow generators to follow the schedule better. Next, we elaborate how the online power balancing scheme shown in Figure 3.3 works under this new market structure step by step.

**Step 1: Bidding:** In the bidding stage prior to RTD, slow generator $i$ submits its "simple bid" function $C^s_{S,i}(P_i)$, or equivalently, supply curve $P_i(\lambda)$, while fast generator $j$ provides its conventional bid function $C_{F,j}(P_j)$ or supply curve $P_j(\alpha)$. $\lambda$ and $\alpha$ are RTD and redispatch prices, respectively.

**Step 2: Risk-based real-time dispatch:** Before the 5-min interval begins, the (I)SO dispatches slow generators and clears the RTD price $\lambda_t$ by solving the following economic dispatch problem with simple bids of slow generators $C^s_{S,i}(P_{i,t})$:

$$\min_{P_{i,t}, i \in \mathcal{S}} \quad \sum\nolimits_{i \in \mathcal{S}} C^s_{S,i}(P_{i,t}) \tag{3.13}$$

$$s.t. \quad \sum\nolimits_{i \in \mathcal{S}} P_{i,t} = P_D \tag{3.14}$$

$$|F| \leq \overline{F} \tag{3.15}$$

$$|P_{i,t} - P_{i,t-1}| \leq RR_i \tag{3.16}$$

$$P_i \in [\underline{P}_i, \overline{P}_i] \tag{3.17}$$

The RTD price $\lambda_t$ is defined as the corresponding locational marginal prices (LMP) calculated by solving the real-time dispatch problem. At this point, the *demand* pays $\lambda_t P_D$, while the *slow generator* $i$ is paid by $\lambda_t P_{i,t}$. The market-based RTD is computationally much simpler comparing to the centralized risk-based RTD (3.1)-(3.6), since the proposed market structure distributes the complexity of risk management to the simple bids of individual market participants.

**Step 3: Redispatch:** Once the actual deviations $\Delta P_{i,t}$ are measured at the beginning of the 5-min interval, the (I)SO redispatches fast generators to compensate the imbalances by solving the re-dispatch problem (3.8)-(3.12), the same as in the centralized framework. The redispatch price $\alpha_t$ is defined as the corresponding LMP in the re-dispatch problem. Different from RTD, in redispatch stage the slow generators becomes inelastic demand because they need fast generators to compensate their *actual* deviations $\Delta P_{i,t}$. In this stage, the *slow generator* $i$ pays $\alpha_t \Delta P_{i,t}$ while the *fast generator* $j$ gets paid $\alpha_t P_{j,t}$ for compensating the slow generators' deviations.

### 3.4.2 Profit Maximization of Generators under Proposed Market Structure

Next, we explain how slow and fast generators create their "simple bids" by internalizing potential risk of deviations and the corresponding costs under the new market structure.

**Slow generators:** Since slow generators are charged according to their deviations from scheduled generation, they must consider the cost associated with the deviations when they create their bids. However, the cost of deviations $\alpha_t \Delta P_{i,t}$, as well as generation cost $C_{S,i}(P_{i,t} - \Delta P_{i,t})$, are stochastic, because neither the redispatch price $\alpha_t$ nor its actual deviation $\Delta P_{i,t}$ is known to the slow generator $i$ at the bidding stage. Instead, it only knows the probability distribution of $\alpha_t$ and $\Delta P_{i,t}$. To this end, we introduce the concept of *conditional value-at-risk* to formulate these stochastic costs.

The slow generator $i$'s profit maximization internalizing the potential deviations is as follows:

$$\max_{P_{i,t}} \quad \lambda_t P_{i,t} - CVaR_{\beta_i}\left(C_{S,i}(P_{i,t} - \Delta P_{i,t}) + \alpha_t \Delta P_{i,t}\right) \tag{3.18}$$

$$s.t. \quad |P_{i,t} - P_{i,t-1}| \leq RR_i \tag{3.19}$$

$$P_{i,t} \in [\underline{P}_i, \overline{P}_i] \tag{3.20}$$

$$\alpha_t \sim f(\alpha_t), \quad \Delta P_{i,t} | P_{i,t} \sim f(\Delta P_{i,t}) \tag{3.21}$$

where $\beta_i \in [0,1]$ the risk preference of slow generator $i$ as previously explained in Section III. For slow generator $i$ with risk preference $\beta_i$, out of the profit maximization is the optimal generation output $P^*_{i,t}$ given RTD price $\lambda_t$. By varying $\lambda_t$, a supply curve $P^*_{i,t}(\lambda_t)$ as well as the "simple bid" $C^s_{S,i}(P_{i,t})$ are created.

**Fast generators:** For fast generators, the bidding process is much more straight-forward: since they exactly follow the (I)SO's command and have no deviations, they are not charged with corresponding costs; Instead, they only participate re-dispatch market with conventional cost/bid function $C_{F,j}(P_j)$.

### 3.4.3 Comparison with Centralized Approach

The market-based approach differs from centralized approach in various ways. In terms of information flow between generators and the (I)SO, the market-based approach allows each generator to internalize the statistical information of its own deviations into simple bids, while in centralized approach this information must be obtained by the (I)SO by statistical approaches. In terms of risk preferences, different generators can have their own risk measures and preferences in the market-based approach, while in the centralizd approach there is a risk preference common to all.

While market-based approach and centralized approach are fundamentally different, we find that they are connected in the sense that, under certain conditions, centralized and market-based approaches lead to the same results. Based on decomposition theory [27], we have the following theorems as a theoretical validation of our market design.

**Theorem 2.** *If expected cost is considered and $\Delta P_i$ are sufficiently small comparing to $P_i$ for all slow generators, the market-based dispatch and centralized risk-based dispatch yields the same solution.*

*Proof.* We start from the centralized risk-based dispatch problem where expected generation cost is minimized.

$$\begin{aligned} \min_{P_{i,t}, i \in \mathcal{S}} \quad & \mathbb{E}\left(\sum_{i \in \mathcal{S}} C_{S,i}(P_{i,t} - \Delta P_{i,t}) + R(\Delta P_t)\right) \\ s.t. \quad & \textstyle\sum_{i \in \mathcal{S}} P_{i,t} = P_D \\ & \text{Constraints (3.3) - (3.6)} \end{aligned}$$

The recovery cost can be written as Taylor expansions:

$$R(\Delta P_t) = \sum_i \frac{\partial R}{\partial \Delta P_{i,t}} \Delta P_{i,t} + O(\|\Delta P_t\|_2^2)$$

The objective function can then be written as:

$$\begin{aligned}
&\mathbb{E}\left(\sum_{i\in\mathcal{S}} C_{S,i}(P_{i,t} - \Delta P_{i,t}) + R(\Delta P_t)\right) \\
=&\mathbb{E}\left(\sum_{i\in\mathcal{S}} C_{S,i}(P_{i,t} - \Delta P_{i,t}) + \sum_i \frac{\partial R}{\partial \Delta P_{i,t}} \Delta P_{i,t} + O(\|\Delta P_t\|_2^2)\right) \\
=&\sum_{i\in\mathcal{S}} \mathbb{E}\left(C_{S,i}(P_{i,t} - \Delta P_{i,t}) + \frac{\partial R}{\partial \Delta P_{i,t}} \Delta P_{i,t}\right) + \mathbb{E}O(\|\Delta P_t\|_2^2)
\end{aligned}$$

where $\frac{\partial R}{\partial \Delta P_{i,t}} = \alpha_i$ is the redispatch LMP. The only couping among different generators is the term $\mathbb{E}O(\|\Delta P_t\|_2^2)$. When $\Delta P_i$ are sufficiently small comparing to $P_i$ for all slow generators, $\mathbb{E}O(\|\Delta P_t\|_2^2) \approx 0$. In uncongested cases, we can write the lagrangian of the optimization problem:

$$\begin{aligned}
L(P_{i,t}, \lambda) =& \sum_{i\in\mathcal{S}} \mathbb{E}\left(C_{S,i}(P_{i,t} - \Delta P_{i,t}) + \frac{\partial R}{\partial \Delta P_{i,t}} \Delta P_{i,t}\right) + \lambda\left(P_D - \sum_{i\in\mathcal{S}} P_{i,t}\right) \\
=& \lambda P_D - \sum_i \left(\lambda P_{i,t} - \mathbb{E}\left(C_{S,i}(P_{i,t} - \Delta P_{i,t}) + \frac{\partial R}{\partial \Delta P_{i,t}} \Delta P_{i,t}\right)\right)
\end{aligned}$$

Given that the second term is separable among generators, according to dual decomposition theory, if original problem is convex, minimization of the lagrangian can be achieved by all generators solving their own profit maximization problem:

$$\begin{aligned}
\max_{P_{i,t}} \quad & \lambda_t P_{i,t} - \mathbb{E}\left(C_{S,i}(P_{i,t} - \Delta P_{i,t}) + \alpha_t \Delta P_{i,t}\right) \\
s.t. \quad & \text{Constraint (3.19) - (3.21)}
\end{aligned}$$

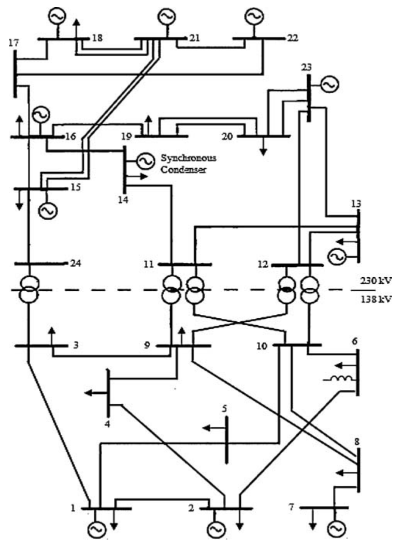


Figure 3.4: The IEEE 24-bus reliability test system (IEEE One Area RTS-96)

where $\alpha_t$ is the redispatch price. $\square$

## 3.5 Evaluation

### 3.5.1 Setup

**Evaluation framework:** We employ a custom Matlab-based simulation framework in the IEEE 24-bus reliability test system (IEEE One Area RTS-96) [39] as shown in Figure 3.4. We add the profiles of slow generators in Table 3.1 to the existing set of generators in [39]. We compare the proposed risk-based market structure with conventional RTD, where generators are paid based on their RTD schedules and prices without considering risks of deviations. We also implemented the generator-side and (I)SO-side decision making in Smart Grid in a Room Simulator (SRGS).

**Resource allocation schemes:** We compare the following schemes:

1. *No deviation*: This is the baseline case where slow generators do not have deviations, i.e., they can follow exactly the system operator's command. We use this result as a lower bound of the generation cost and normalize the cost of other schemes.

2. *Conventional*: The conventional real-time dispatch scheme that solves an DC-OPF problem to schedule slow generators, without considering the possible deviations. At re-dispatch stage, slow generators' deviations are compensated by solving another DC-OPT problem on fast generators.

3. *Risk-based*: The risk-based approach entails solving a DC-OPF problem on (I)SO side given slow generators' bid function internalizing the risk of deviations. The re-dispatch stage is similar to conventional dispatch, i.e., schedule fast generators to compensate deviations of slow generators.

**Metrics:** We evaluate the algorithms using the following performance metrics:

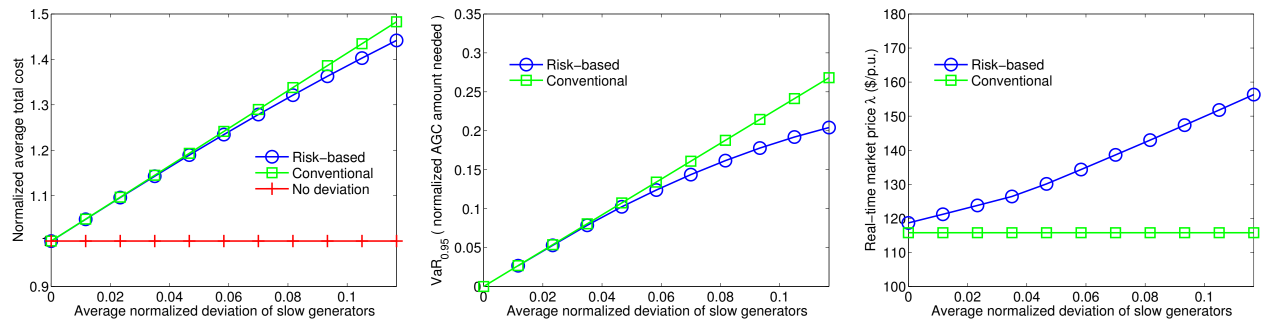
1. *Total generation cost*: Sum of actual generation cost of both slow and fast generators in the actual 5-min interval.

2. *AGC amount needed*: The 95% VaR of AGC amount needed to cover slow generators' deviations if no fast generators are used to compensate the imbalances.

3. *Real-time market price*: The *ex-ante* average locational marginal price of the real-time dispatch stage. This can be techinically calculated by the langrangian multipliers of the corresponding constraints in the real-time dispatch problems.

### 3.5.2 System-Wide Impact

First, we show end-to-end result of proposed risk-based dispatch in Figure 3.5. We focus on the following system-wide impact of proposed approach: total generation cost, AGC amount needed, locational marginal prices, and cost allocation to generators.



(a) (b) (c)

Figure 3.5: System-wide impact of deviations

**Total generation cost:** Figure 3.5a shows the comparison of total generation costs under conventional RTD and the risk-based market rule. As slow generators' deviations are increasing, the total generation cost of conventional RTD is significantly increased. This is because conventional RTD does not consider the potential deviations of slow generators, which may lead to large imbalances that needs more expensive fast generators to compensate. On the other hand, comparing to the conventional RTD, the average total generation cost, as well as the standard deviation of the total cost, are reduced by adopting the risk-based market rule. In addition, the amount of cost reduction becomes more and more significant as deviation is increased.

**AGC requirement:** Figure 3.5b shows the 95% Value-at-Risk of AGC amount needed to cover the deviations of slow generators if no fast generators are scheduled. First, as slow generators' deviation grow larger, more AGC resources are needed to cover their potential deviations in the actual 5-min interval. This confirms our observation that slow generators' deviations lead to increased stress on AGC. However, by using the risk-based approach, the agc requirement is reduced as expected deviation is mitigated. The advantage of risk-based approach becomes larger as deviation grows, resulting in a 20% reduction in AGC amount when average deviation is 10%.

**Prices:** Figure 3.5c shows the real-time market price under the proposed risk-based market rule. The price change is near zero when average deviation of slow generators is zero. As deviation is increasing, the price of risk-based dispatch keeps increasing while the price of conventional dispatch stays unchanged. Note that while increasing, the price of risk-based dispatch reflects the actual marginal cost of electric power: more expensive fast generators are needed to compensate the increased imbalances caused by more deviations. On the other hand, prices of conventional dispatch cannot reflect the cost of recovery, leading to potential variations of *ex-post* prices and large deviations.

**Cost allocation:** In order to understand deeper the reason behind aforementioned system-wide benefits, we also investigate how generators with different deviations are penalized or rewarded differently in the proposed risk-based market structure comparing to conventional RTD. In conventional RTD, generators are dispatched without considering risks of deviations, therefore, the cheaper generators are scheduled to generate more power and obtain more profit. In risk-based approach, generators with larger deviations are penalized comparing to conventional RTD and scheduled to generated less power, so as to avoid potential large imbalances during the actual 5-min interval. On the other hand, generators with less uncertainty are rewarded by being allocated more generation in the upcoming interval, since they follow the (I)SO's command more accurately. As a result, the profit of generators with large deviation reduces, while profit of generators with less uncertainty increases comparing to the conventional RTD. As such, risk-based dispatch allows more fair cost allocation while in conventional dispatch the slow generators causing deviations are not penalized appropriately. Naturally, this will place incentives for slow generators to improve their control accuracy so as to increase profitability. We will further investigate how this will impact individual generators' decision making shortly.

### 3.5.3 Impact on Individual Generators

Next, we investigate how the adoption of risk-based market rule will impact the decision making of individual generators.

**Bid function:** Under risk-based market rule, the slow generators make "simple" bids internalizing the risk of deviations, as is shown in previous sections. Now we zoom in and show in Figure 3.6 an example of bid functions of a slow generators (with 10% expected deviation).

The blue curve shows the bid function when there is no deviation or no charge in the system. Now as the generator can deviate from schedule for 10% in average, it needs to consider the potential charge in risk-based market rule, as its deviation will be charged in redispatch stage. As such, the marginal cost of providing a specific amount of power is increased, and the bid curve moves to the right in the plot. The risk preference $\beta$ also plays an important role in the bid function: As $\beta$ increases, the generator becomes more conservative and considers $CVaR_{\beta}$ of cost in its profit maximization, and therefore perceives higher marginal cost.

**Profit vs deviation:** We study how profit of individual generator changes under the risk-based market rule if its deviation characteristic is improved (not shown). If slow generators can reduce its uncertainty level and become more accurately controllable in real-time, it can get more profit under the proposed risk-based market rule. This result shows that the proposed market rule can facilitate better risk management and less uncertainty level in the slow generators' operations in the long run.

### 3.5.4 Sensitivity Analysis

We conduct sensitivity analysis w.r.t. key system parameters, i.e., risk preferences and redispatch prices.

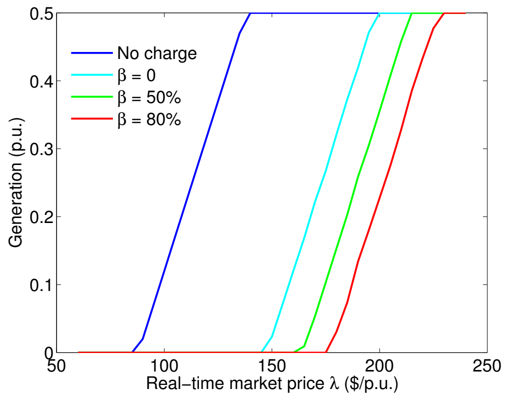
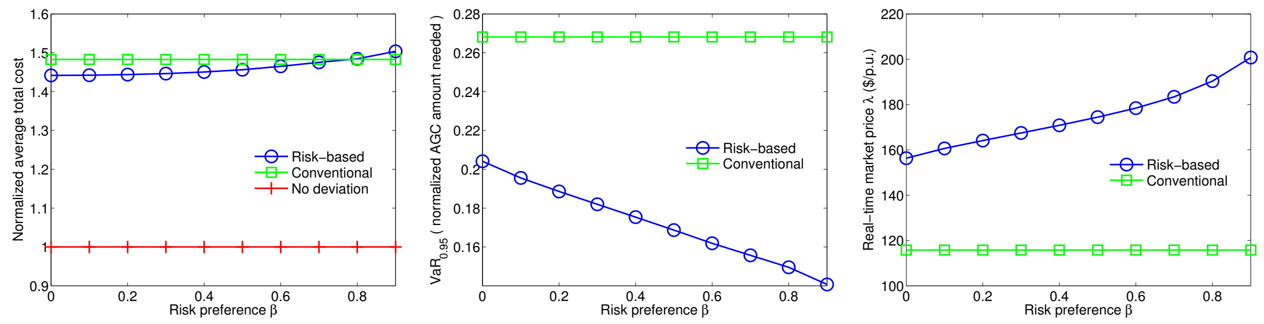


Figure 3.6: Example bid function for slow generators

**Impact of risk preference:** Figure 3.7 shows the system-wide impact of varying risk preferences. Note that the risk perference refers to $\beta$, i.e., the confidence level of the conditional Value-at-Risk of slow generators as they calculate bid functions. $\beta = 0$ means the slow generator seeks to maximize expected profit while $\beta = 1$ corresponds to worst-case profit. While different slow generators can have different risk preferences as is suggested in risk-based market, here for illustration purposes we only show results with $\beta$ being the same for all slow generators.

First, we observe from Figure 3.7a that with risk-based market rule, the total generation cost depends on the risk preference $\beta$: when $\beta$ increases, the generators bid more conservatively in order to avoid large deviation and corresponding charges. As such, the resulting average total generation cost increases, while the variability of the cost decreases (not shown). This leads to an interesting trade-off between the mean and the varibility of the total cost, controlled by the risk preference $\beta$. Note that when $\beta > 0.8$, the slow generators can become over-conservative, making the total generation cost even higher than conventional dispatch. This is because the cost reduction of less deviations is not enough to cover the increased fuel cost of more expensive but non-deviating generators.

Second, as $\beta$ increases, the amount of AGC needed is significantly reduced as shown in Figure 3.7b. When $\beta = 0$, the AGC requirement is reduced for 20+%; As slow generators



(a) (b) (c)

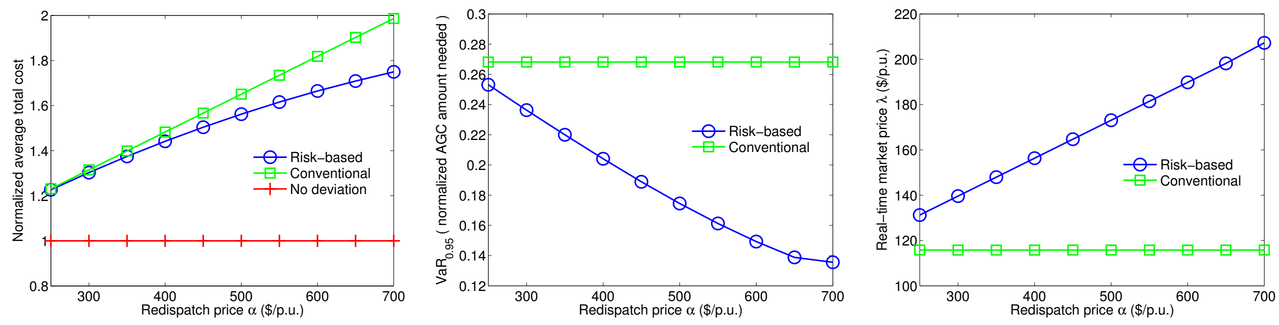
Figure 3.7: Impact of risk preference

become more conservative, the AGC requirement keeps decreasing; At $\beta = 0.9$, only 50% of AGC is needed comparing to conventional dispatch, which is a significant improvement.

We further look at the impact on system prices in Figure 3.7c. As the system level risk preference $\beta$ increases, the slow generators becomes more risk-averse, the RTD price $\lambda$ increases, while the redispatch price $\alpha$ decreases. This is because if slow generators are more risk-averse, the ones with larger uncertainty, even though cheaper, will be scheduled for less generation to avoid potential large deviations, which increases the system price in RTD stage. As a result, as $\beta$ increases, the potential imbalances during the actual interval are reduced, which brings the re-dispatch price $\alpha$ down.

**Impact of re-dispatch price:** Finally, we study how re-dispatch price $\alpha$ impacts the system performance in Figure 3.8. While in general re-dispatch price depends on the actual deviation and can thus be a stochastic variable at the RTD stage, we use a fixed re-dispatch price by setting a linear cost function for all fast generators and assume there is no congestion in re-dispatch stage. We have the following observations:

First, while re-dispatch price does not affect total cost when no slow generators are present, in cases with slow generators the total generation cost are increased as $\alpha$ grows (Figure 3.8a). This is because the cost of fast generators is increased as $\alpha$ increases, resulting



(a) (b) (c)

Figure 3.8: Impact of redispatch price $\alpha$

in increased overall cost. We further observe that the risk-based approach can significantly reduce total generation cost, and the reduction increases as $\alpha$ increases.

Second, as shown in Figure 3.8b, the AGC amount needed is significantly decreased as redispatch price is increased in risk-based approach. The reason is that as redispatch price increases, in order to avoid high deviation charges the slow generators bid more conservatively, resulting in reduced overall expected deviation. On the other hand, as deviation charges are not considered in conventional dispatch, the AGC needed keeps unaffected.

Finally, the re-dispatch price also affects the RTD price $\lambda$: As shown in Figure 3.8c, the RTD prices increase as re-dispatch increase in risk-based approach, as the recovery cost of deviations is reflected in the RTD price. As conventional RTD does not consider recovery cost in RTD stage, the RTD price does not change, but the *ex-post* price can change due to deviations.

### 3.5.5 Summary of Results

Our main findings are summarized as follows:

1. The adoption of proposed risk-based approach has major impact on system-wide performances: Comparing to conventional dispatch, the risk-based dispatch can reduce

expect total generation cost, significantly reduce AGC amount needed, and can reflect the recovery cost in real-time market price.

2. The risk-based market rule allocates cost to the slow generators that causes deviations, and thus allowing slow generators to consider and internaling this risk in the bid functions, and more importantly, providing incentives for slow generators to improve control accuracy and reduce deviations.

3. The sensitivity analysis shows that both risk preferences of slow generators and redispatch price can affect system performance: Larger $\beta$ results in more conservative decisions and reduces AGC amount needed, but leads to increased total cost; Higher redispatch price $\alpha$ increases the advantage of risk-base approach.

# Chapter 4

# Conclusions and Future Work

In this chapter, we summarize the findings in this dissertation and suggest possible future work. We envision that a mathematical theory can significantly improve future resource allocation ecosystems in smart infrastructures.

## 4.1 Concluding Remarks

In this dissertation, we focus on the design of better resource allocation schemes in smart infrastructure that respects practical constraints. Instead of a general theoretical analysis, we adopt a case-study-based approach to provide insights from two representative systems: internet video delivery and electric power entworks.

Specifically, in video delivery networks, we focus on the joint bandwidth allocation and bitrate adaptation problem. We first attempt to improve resource allocation from client-side in a fully distributed way: designing a MPC-based bitrate adaptation algorithms that can dynamically optimize user QoE, essentially regarding available bandwidth as black-boxed input. Given the analysis that TCP is not sufficient to guarantee QoE fairness, we further consider the design of a router-assited network bandwidth allocator that treats players as closed-loop systems and allocate bandwidth according to the QoE function as well as their states.

On the other hand, as an example to adopt market-based resource allocation schemes, we investigate the design and evaluation of the risk-based real-time dispatch and market design. This is motivated by the observation that slow generators' deviation can negatively impact system performance, e.g., adding stress to the AGC. While a centralized risk-based dispatch entails learning each slow generator's behaviour and make dispatch decisions taking into account the possible deviation, a risk-based market rule is proposed to penalize slow generators for their deviations and provide incentives for them to improve control accuracy.

Summing up the findings from the case studies, we have the following insights on the general design of resource allocation schemes: First, there will be more overall benefits for the centralized controller to know and manage each user/agent's dynamics when agents are more heterogeneous. For example, from the case study of multiplayer adaptive video streaming, we know that a router-assisted control can benefit more if players have more diversed buffer levels and QoE functions. Second, while a centralized controller can implicitly know the characteristics of each agent, it may in turn be beneficial to design a mechanism that provides incentives to shape agent's behavior. For example, in case studies in electic power networks, risk-based market rule can provide incentives and leave the complexity to each slow generator; while in a centralized scheme slow generators are not charged according to deviations.

## 4.2 Future Work

While we focus on a range of aspects in case studies in video delivery and electric power networks, we acknowledge that there are several limitations in the current work. As such, we suggest possible and interesting future work in this direction. In general, the current work can be improved by 1) improved modeling using real-world data and 2) evaluation in real-world scenario.

**Single-player adaptive video streaming:** While we evaluate the proposed MPC algorithm in real video players, we have not carried out experiments in real networks or studied real improvement on user experience. There are several factors that makes the real evaluation results potentially useful. First, it helps us understand and quantify the improvement of user QoE under real environment and will give insights to the industry on whether MPC-based players should be employed in practice. Second, we can study the robustness of MPC under multiplayer scenario by running MPC with players with other adaptation strategies and cross traffic. As such, in future work, we plan to evaluate MPC algorithms in real networks instead of emulated networks.

In addition, we want to explore the possibility to further reduce FastMPC table size by using an active-learning-based approach. We observe from our experiments that, the actual adaptation policy using MPC is highly structured, i.e., a lot of entries in FastMPC table are the same given similar scenarios. As such, we expect the adaptation policy computed by MPC can be expressed by a simpler function form instead of using a look-up table, and therefore can be potentially generated using active-learning-based approaches.

**Multi-player adaptive video streaming:** First, we want to apply decomposition and distributed optimization theory to design distributed schemes with information exchange for joint bandwidth allocation and bitrate adaptation, i.e., how each entity (players and router) exchanges information and makes local decisions so that the system as a whole achieves desired optimality or fairness. Given that the joint optimization problem is non-convex, we plan to convexify it by linearizing the buffer dynamic equations.

Second, our current model of unideal TCP-based bandwidth allocation is based on the measurement from Huang et al. [42]. In the future work, we want to conduct real network measurement and build a data-driven model to better characterize the bandwidth allocation function $h(\cdot)$ in different network conditions so as to analyze the convergence to fairness in practice. Based on the data-driven model of $h(\cdot)$, we also want to extend our steady-state analysis and simulation to heteogeneous player cases and consider more general RB, BB and hybrid policies with more general throughput prediction model.

Finally, we want to evaluate the proposed router-assisted algorithms in practice, especially in the case of heteogeneous players. We want to explore 1) the interaction of players with the same control algorithms but different throughput prediction algorithms, 2) the interaction of players with different control algorithms. While conducting real experiments, we will also explore the feasibility of learning players' state and adaptation policies at router side.

**Risk-based real-time dispatch:** In future work, we will first use real data to develop a data-driven model of slow generators' deviations. In this thesis we assume the deviations follow Gaussian distribution and cannot be controlled by the generators, however, in practice, this assumption may not hold any more: Deviations may depend on a variety of factors, such as ramp up/down amount, time of day, etc.; Also, the deviations may be due to the slow generators' strategic decisions in the scenario of gaming. We want to use real bid data from (I)SOs to learn a model of the deviations and verify our simulation results and corresponding insights.

Second, we want to further study market dynamics and strategic behavior as a result of the risk-based market design. The proposed risk-based market design can significantly change the behavior of slow and fast generators in the following aspects: 1) In the long term, based on the profitability in real-time dispatch and re-dispatch markets, the number of slow/fast generators/AGC resources can change accordingly; 2) In the short term, based on the prices in the two market, fast generators may allocate their capacity to participate real-time dispatch and re-dispatch market; 3) In the short term, the slow/fast generators participating in both market stages may strategically change their deviations by anticipating the system operator's actions and market prices. In future work, we want to study and quantify these impact and their implications on the market design.

Finally, in future work we want to implement the proposed market mechanism and conduct large-scale distributed simulation in Smart Grid in a Room Simulator (SGRS). SGRS provides a distributed, class-oriented simulation environment in Matlab that enables large-scale emulation of electricity markets. While in our simulation we are able to show the improvement in controlled simulations, what is still lacking is a show-case of how this market mechanism will perform in practical environments where generators do not have perfect prediction of prices and thus submit approximate, piece-wise linear bid functions. Large-scale emulations in SGRS will provide us with a better understanding of the implications on real-world scenarios.

## 4.3 Toward a Mathematical Theory for Better Resource Allocation Ecosystems

While this thesis aims at developing principled resource allocation schemes that respect practical constraints, it is only utilizing the limited sensing and control capability at the edge of the networks, which is constrained by the structure of the *resource allocation ecosystems*, namely, entities that owns the resources/capabilities and the way these entities interact with each other. For example, players of Netflix cannot communicate with or control players with other video providers and non-video applications. As we observe from this thesis, the structure of the resource allocation ecosystems, if not properly designed, will pose significant limitations and challenges for the design of optimal resource allocation schemes. As such, while in the short term it can benefits the system to design resource allocation schemes that works practically in the current ecosystem, it becomes increasingly critical in the long term to rethink and design better resource allocation ecosystems.

As an illustrating example, in video delivery networks, there are a variety of entities that controls different parts of the network and optimize their own profits. For example, video source providers such as Netflix and YouTube, own the players and the algorithms inside and want to maximize the user-perceived QoE; Content delivery networks (CDN), such as Akamai and Level3, assign players to content servers and optimize the a combination of metrics for general web traffic; Internet service providers (ISP), such as Comcast and Verizon, can control the bandwidth available to end players/servers and want to reduce the operating cost while provide guaranteed bandwidth according to agreement with users. The maximization of users' QoE needs the coordination of all entities, however, as the current resource allocation ecosystems evolve in an ad-hoc manner without principled design, the result of the current coordination schemes is still far from optimal.

A lot of recent research (e.g., [46]) has argued the need for better resource allocation ecosystems, and proposed initial solutions that tackles specific resource allocation problems. However, there are two key questions that remained unanswered by prior research:

1. Is the structure of the current resource allocation ecosystem sufficient to develop close-to-optimal resource allocation schemes? For instance, does it result in instability or suboptimal resource allocation if each entity optimize their own objective?

2. Can we design structure of the ecosystems to achieve better resource allocation? For example, what is the information that needs to be exchanged between different entities to improve overall allocation?

To answer these questions, we envision that a principled model-based approach will be helpful in the analysis and design of the better ecosystems. To this end, we plan to develop a mathematical theory for better resource allocation ecosystems.

In particular, we plan to model the ecosystems in two time scales: In the faster time scale, we model each entity as a dynamical system with a feedback controller based on the actual information exchange patterns. Each entity is able to control their own input based on its own state variable and the observed output from other entities. In the slower time scale, we model each entity as an agent that makes discrete decisions that optimize its objective. Given the stochastic nature of the network bandwidth and user's demand, we also plan to develop a data-driven dynamical model of the evolution of networks and demand. First, we want to analyze the existing ecosystem through control theory and distributed optimization/decomposition theory and derive results on the stability and optimality. Based on the insights from the analysis, we want to develop better protocols and information exchange patterns between the entities to enable better resource allocation and drive critical change in smart infrastructures.

# Bibliography

[1] Dash-Industry-Forum, dash.js . https://github.com/Dash-Industry-Forum/dash.js/wiki.

[2] Adobe HTTP Dynamic Streaming. www.adobe.com/products/hds-dynamic-streaming.html.

[3] Adobe OSMF player. http://www.osmf.org.

[4] Akamai HD network. www.akamai.com/hdnetwork.

[5] Apple's HTTP Live Streaming. https://developer.apple.com/streaming/.

[6] DASH-264 JavaScript reference client landing page 1.4.0. http://dashif.org/reference/players/javascript/1.4.0/samples/dash-if-reference-player/index.html.

[7] DASH Industry Forum members. http://dashif.org/members/.

[8] DASH VLC plugin. http://www-itec.uni-klu.ac.at/dash/?page_id=10.

[9] FCC dataset. https://www.fcc.gov/measuring-broadband-america. Accessed: 2014-12-01.

[10] Google OnHub router. https://on.google.com/hub/.

[11] HSDPA dataset. `http://home.ifi.uio.no/paalh/dataset/hsdpa-tcp-logs`. Accessed: 2014-12-01.

[12] Netflix. `http://www.netflix.com/`.

[13] OSMF 2.0 release code. `http://sourceforge.net/projects/osmf.adobe/files/latest/download`.

[14] Programmable OpenWrt. `https://wiki.openwrt.org/doc/uci/network`.

[15] Smooth Streaming protocol. `http://go.microsoft.com/?linkid=9682896`.

[16] The demo page for our MPC-based bitrate adaptation. `http://users.ece.cmu.edu/~vsekar/mpcdash.html`.

[17] YouTube live encoder settings, bitrates and resolutions. `https://support.google.com/youtube/answer/2853702?hl=en`.

[18] I. Sodagar. The MPEG-DASH Standard for Multimedia Streaming Over the Internet. *IEEE Multimedia*, 2011.

[19] S. Akhshabi, L. Anantakrishnan, A. C. Begen, and C. Dovrolis. What happens when http adaptive streaming players compete for bandwidth? In *Proceedings of the 22Nd International Workshop on Network and Operating System Support for Digital Audio and Video*, NOSSDAV '12, pages 9–14, New York, NY, USA, 2012. ACM.

[20] S. Akhshabi, L. Anantakrishnan, C. Dovrolis, and A. C. Begen. What Happens when HTTP Adaptive Streaming Players Compete for Bandwidth? In *Proc. NOSSDAV*, 2012.

[21] S. Akhshabi, L. Anantakrishnan, C. Dovrolis, and A. C. Begen. Server-based traffic shaping for stabilizing oscillating adaptive streaming players. In *Proceeding of the*

*23rd ACM Workshop on Network and Operating Systems Support for Digital Audio and Video*, pages 19–24. ACM, 2013.

[22] S. Akhshabi, L. Ananthakrishnan, A. Begen, and C. Dovrolis. Server-Based Traffic Shaping for Stabilizing Oscillating Adaptive Streaming Players. In *Proc. ACM SIGMM NOSSDAV*, 2013.

[23] A. Balachandran, V. Sekar, A. Akella, S. Seshan, I. Stoica, and H. Zhang. Developing a Predictive Model of Quality of Experience for Internet Video. In *Proc. ACM SIGCOMM*, 2013.

[24] A. Beben, P. Wiśniewski, J. M. Batalla, and P. Krawiec. Abma+: lightweight and efficient algorithm for http adaptive streaming. In *Proceedings of the 7th International Conference on Multimedia Systems*, page 2. ACM, 2016.

[25] D. P. Bertsekas, D. P. Bertsekas, D. P. Bertsekas, and D. P. Bertsekas. *Dynamic Programming and Optimal Control*, volume 1. Athena Scientific Belmont, MA, 1995.

[26] D. Bertsimas, E. Litvinov, X. A. Sun, J. Zhao, and T. Zheng. Adaptive robust optimization for the security constrained unit commitment problem. *Power Systems, IEEE Transactions on*, 28(1):52–63, 2013.

[27] S. Boyd and L. Vandenberghe. *Convex optimization.* Cambridge university press, 2009.

[28] E. F. Camacho and C. B. Alba. *Model Predictive Control.* Springer, 2013.

[29] M. Chiang. *Nonconvex Optimization for Communication Networks*, pages 137–196. Springer US, Boston, MA, 2009.

[30] L. D. Cicco, V. Caldaralo, V. Palmisano, and S. Mascolo. TAPAS: a Tool for rApid Prototyping of Adaptive Streaming algorithms. In *Proc. CoNext VideoNext workshop*, 2014.

[31] G. Cofano, L. De Cicco, T. Zinner, A. Nguyen-Ngoc, P. Tran-Gia, and S. Mascolo. Design and experimental evaluation of network-assisted strategies for http adaptive streaming. In *Proceedings of the 7th International Conference on Multimedia Systems*, MMSys '16, pages 3:1–3:12, New York, NY, USA, 2016. ACM.

[32] S.-J. Deng and S. S. Oren. Electricity derivatives and risk management. *Energy*, 31(6):940–953, 2006.

[33] F. Dobrian, V. Sekar, A. Awan, I. Stoica, D. A. Joseph, A. Ganjam, J. Zhan, and H. Zhang. Understanding the Impact of Video Quality on User Engagement. In *Proc. ACM SIGCOMM*, 2011.

[34] G. F. Franklin, J. D. Powell, and M. L. Workman. *Digital Control of Dynamic Systems*, volume 3. Addison-wesley Menlo Park, 1998.

[35] A. Ganjam, F. Siddiqui, J. Zhan, X. Liu, I. Stoica, J. Jiang, V. Sekar, and H. Zhang. C3: Internet-Scale Control Plane for Video Quality Optimization. In *Proc. NSDI*, 2015.

[36] P. Georgopoulos, Y. Elkhatib, M. Broadbent, M. Mu, and N. Race. Towards network-wide qoe fairness using openflow-assisted adaptive video streaming. In *Proceedings of the 2013 ACM SIGCOMM Workshop on Future Human-centric Multimedia Networking*, FhMN '13, pages 15–20, New York, NY, USA, 2013. ACM.

[37] M. Ghobadi, Y. Cheng, A. Jain, and M. Mathis. Trickle: Rate Limiting YouTube Video Streaming. In *Proc. USENIX ATC*, 2012.

[38] S. Gouache, G. Bichot, A. Bsila, and C. Howson. Distributed and Adaptive HTTP Streaming. In *Proc. ICME*, 2011.

[39] C. Grigg, P. Wong, P. Albrecht, R. Allan, M. Bhavaraju, R. Billinton, Q. Chen, C. Fong, S. Haddad, S. Kuruganty, W. Li, R. Mukerji, D. Patton, N. Rau, D. Reppen, A. Schnei-

der, M. Shahidehpour, and C. Singh. The ieee reliability test system-1996. a report prepared by the reliability test system task force of the application of probability methods subcommittee. *IEEE Transactions on Power Systems*, 14(3):1010–1020, Aug 1999.

[40] D. Havey, R. Chertov, and K. Almeroth. Receiver Driven Rate Adaptation for Wireless Multimedia Applications. In *Proc. MMSys*, 2012.

[41] R. Houdaille and S. Gouache. Shaping HTTP Adaptive Streams for a Better User Experience. In *Proc. MMSys*, 2012.

[42] T.-Y. Huang, N. Handigol, B. Heller, N. McKeown, and R. Johari. Confused, Timid, and Unstable: Picking a Video Streaming Rate is Hard. In *Proc. IMC*, 2012.

[43] T.-Y. Huang, R. Johari, N. McKeown, M. Trunnell, and M. Watson. A Buffer-Based Approach to Rate Adaptation: Evidence from a Large Video Streaming Service. In *Proc. ACM SIGCOMM*, 2014.

[44] M. D. Ilić, X. Yin, Q. Liu, and Y. Weng. Toward combining intra-real time dispatch (RTD) and AGC for on-line power balancing. In *Power and Energy Society General Meeting, 2014.*, pages 1–5. IEEE, July 2014.

[45] R. Jain, D.-M. Chiu, and W. R. Hawe. *A quantitative measure of fairness and discrimination for resource allocation in shared computer system*, volume 38. Eastern Research Laboratory, Digital Equipment Corporation Hudson, MA, 1984.

[46] J. Jiang, X. Liu, V. Sekar, I. Stoica, and H. Zhang. Eona: Experience-oriented network architecture. In *Proceedings of the 13th ACM Workshop on Hot Topics in Networks*, page 11. ACM, 2014.

[47] J. Jiang, V. Sekar, and H. Zhang. Improving Fairness, Efficiency, and Stability in HTTP-based Adaptive Video Streaming with FESTIVE. In *Proc. CoNext*, 2012.

[48] J.-Y. Joo and M. Ilić. Multi-temporal risk minimization of adaptive load management in electricity spot markets. In *Innovative Smart Grid Technologies (ISGT Europe), 2011 2nd IEEE PES International Conference and Exhibition on*, pages 1–7. IEEE, 2011.

[49] J.-Y. Joo and M. D. Ilić. Multi-layered optimization of demand resources using lagrange dual decomposition. *IEEE Transactions on Smart Grid*, 4(4):2081–2088, 2013.

[50] S. S. Krishnan and R. K. Sitaraman. Video Stream Quality Impacts Viewer Behavior: Inferring Causality using Quasi-Experimental Designs. In *Proc. IMC*, 2012.

[51] R. Kuschnig, I. Kofler, and H. Hellwagner. Evaluation of HTTP-based Request-Response Streams for Internet Video Streaming. *Multimedia Systems*, pages 245–256, 2011.

[52] L. De Cicco, S. Mascolo, and V. Palmisano. Feedback Control for Adaptive Live Video Streaming. In *Proc. of ACM Multimedia Systems Conference*, 2011.

[53] T. Lan, D. Kao, M. Chiang, and A. Sabharwal. An axiomatic theory of fairness in network resource allocation. In *INFOCOM, 2010 Proceedings IEEE*, pages 1–9, March 2010.

[54] Z. Li, X. Zhu, J. Gahm, R. Pan, H. Hu, A. Begen, and D. Oran. Probe and Adapt: Rate Adaptation for HTTP Video Streaming at Scale. *Selected Areas in Communications, IEEE Journal on*, 32(4):719–733, 2014.

[55] C. Liu, I. Bouazizi, and M. Gabbouj. Parallel Adaptive HTTP Media Streaming. In *Proc. ICCCN*, 2011.

[56] H. Liu, Y. Wang, Y. R. Yang, A. Tian, and H. Wang. Optimizing Cost and Performance for Content Multihoming. In *Proc. ACM SIGCOMM*, 2012.

[57] X. Liu, F. Dobrian, H. Milner, J. Jiang, V. Sekar, I. Stoica, and H. Zhang. A Case for a Coordinated Internet Video Control Plane. In *Proc. ACM SIGCOMM*, 2012.

[58] A. Mansy, M. Fayed, and M. Ammar. Network-layer fairness for adaptive video streams. In *IFIP Networking Conference (IFIP Networking), 2015*, pages 1–9, May 2015.

[59] J. Mo and J. Walrand. Fair end-to-end window-based congestion control. *IEEE/ACM Transactions on Networking (ToN)*, 8(5):556–567, 2000.

[60] R. K. P. Mok, X. Luo, E. W. W. Chan, and R. K. C. Chang. QDASH: A QoE-aware DASH system. In *Proc. MMSys*, 2012.

[61] C. Mueller, S. Lederer, J. Poecher, and C. Timmerer. Libdash - An Open Source Software Library for the MPEG-DASH Standard. In *Proc. ICME*, 2013.

[62] S. S. Oren. Economic inefficiency of passive transmission rights in congested electricity systems with competitive generation. *The Energy Journal*, pages 63–83, 1997.

[63] A. Papavasiliou, S. S. Oren, and R. P. O'Neill. Reserve requirements for wind power integration: A scenario-based stochastic programming framework. *IEEE Transactions on Power Systems*, 26(4):2197–2206, 2011.

[64] L. Popa, A. Ghodsi, and I. Stoica. HTTP as the Narrow Waist of the Future Internet. In *Proc. HotNets*, 2010.

[65] R. Rejaie and J. Kangasharju. Mocha: A Quality Adaptive Multimedia Proxy Cache for Internet Streaming. In *Proc. NOSSDAV*, 2001.

[66] R. T. Rockafellar and S. Uryasev. Optimization of conditional value-at-risk. *Journal of risk*, 2:21–42, 2000.

[67] S. Akhshabi, A. Begen, C. Dovrolis. An Experimental Evaluation of Rate Adaptation Algorithms in Adaptive Streaming over HTTP. In *Proc. MMSys*, 2011.

[68] Y. Sun, X. Yin, J. Jiang, V. Sekar, F. Lin, N. Wang, T. Liu, and B. Sinopoli. Cs2p: Improving video bitrate selection and adaptation with data-driven throughput prediction. In *Proc. of ACM SIGCOMM.* ACM, 2016.

[69] G. Tian and Y. Li. Towards Agile and Smooth Video Adaption in Dynamic HTTP Streaming . In *Proc. CoNext*, 2012.

[70] Y. Wang and S. Boyd. Fast Model Predictive Control using Online Optimization. *Control Systems Technology, IEEE Transactions on*, 18(2):267–278, 2010.

[71] B. White, J. Lepreau, L. Stoller, R. Ricci, S. Guruprasad, M. Newbold, M. Hibler, C. Barb, and A. Joglekar. An Integrated Experimental Environment for Distributed Systems and Networks. *Proc. OSDI*, 2002.

[72] L. Xie, P. M. Carvalho, L. A. Ferreira, J. Liu, B. H. Krogh, N. Popli, and M. D. Ilic. Wind integration in power systems: Operational challenges and possible solutions. *Proceedings of the IEEE*, 99(1):214–232, 2011.

[73] X. Yin, M. D. Ilić, and B. Sinopoli. Toward design of risk-based real-time dispatch at value. In *Innovative Smart Grid Technologies Conference (ISGT), 2015 IEEE Power & Energy Society*, pages 1–5. IEEE, 2015.

[74] X. Yin, A. Jindal, V. Sekar, and B. Sinopoli. A control-theoretic approach for dynamic adaptive video streaming over http. In *Proceedings of the 2015 ACM Conference on Special Interest Group on Data Communication*, pages 325–338. ACM, 2015.

[75] X. Yin, V. Sekar, and B. Sinopoli. Toward a principled framework to design dynamic adaptive streaming algorithms over http. In *Proceedings of the 13th ACM Workshop on Hot Topics in Networks*, page 9. ACM, 2014.

[76] B. Zhang, R. Rajagopal, and D. Tse. Network risk limiting dispatch: Optimal control and price of uncertainty. *IEEE Transactions on Automatic Control*, 59(9):2442–2456, 2014.

[77] Y. Zhang and N. Duffield. On the Constancy of Internet Path Properties. In *IMW*, 2001.