MODELING AND ANALYSIS OF PEER-TO-PEER (P2P) LIVE VIDEO STREAMING

BY

KUANG XU

THESIS

Submitted in partial fulfillment of the requirements
for the degree of Bachelor of Science in Electrical Engineering
in the College of Engineering of the
University of Illinois at Urbana-Champaign, 2009

Urbana, Illinois

Adviser:

Professor Bruce Hajek

Abstract

Peer-to-Peer (P2P) live streaming has become a popular means of distributing real-time online video contents. The distributed nature of the system provides great flexibility, scalability and robustness. The central theme of this thesis revolves around the following general questions that concern the modeling, performance, and stability of P2P live-streaming systems.:

1. What models are effective in describing and analyzing the dynamics of a P2P streaming network and what are the trade-offs in different modeling methods? We examine three stochastic model candidates (Chapter 2) that depict the network at various levels of granularity [1, 2]. While we do not present any explicit results related to the models, they serve not only as conceptual frameworks for understanding the critical characteristics of the network, but also important analytical tools on which later results are based.

2. What are the decision variables involved in the process, and how can we utilize them to optimize the peers' viewing experiences? In particular, we identify the average downloading rate achieved by the network as one of the most important performance measures, as it assesses the video playback continuity experienced by an average peer. We present a heuristic criteria for optimizing permutation-based downloading policies, which outperforms the best Mixed policy in [1].

3. Is the P2P video streaming network stable under certain downloading policies and incentive strategies? Even if the model suggests the *existence* of a healthy steady state with a desirable throughput, is there still a chance for the network to become *stuck* in a state with poor performance? We show uniqueness of marginal distribution for several typical permutation-based downloading policies, indicating the steady-state marginal chunk distributions for these policies are stable. We also show that there exists a bistability of fixed point of marginal distribution when the tit-for-tat incentive requirement is enforced, leading to drastically different continuity performances depending on the initial state of the network.

4. Lastly, we ask what downloading policies a selfish peer would choose in order to maximize its own playback continuity, given that it has complete information of the empirical distribution of other peers. This sheds light on the robustness of downloading policies against malicious peers. We present a necessary condition for any optimal downloading policy.

Acknowledgment

I have been very fortunate to work under the supervision of Professor Bruce Hajek during the course of my undergraduate thesis research. There are simply too many things that I would like to thank Dr. Hajek for, from teaching me hand-to-hand detailed mathematical techniques, to inspiring discussions of research in general. Looking back, from a third-year undergraduate student who naively claimed interests in "something having to do with random systems", to a graduating senior who is getting ready for a journey of engineering research in graduate school, I have learned so much from my advisor over the past year. While many questions for which I set out remain unsolved, and the frustration of hitting a dead end in investigation may have constituted a decent portion of my experience, I have really come to appreciate the nature of theoretical research, its challenges and yet its wonderful rewards in both real-world impacts and intellectual enlightenment. In that regard, Dr. Hajek has taught me many invaluable lessons on the intricate nature of real research and the blunt fact that we are dealing with problems for which no one in the world knows the answers; yet the absence of an answer for an important question is precisely what drives our pursuit.

I would also like to thank Professor Dah Ming Chiu from the Chinese University of Hong Kong for the interesting works from his group on peer-to-peer live streaming, as well as inspiring discussions with us on the topic.

Table of Contents

Chaper 1

Introduction

The world has witnessed a surging demand for videos distributed through the Internet. In many cases, instead of playing the video after downloading the complete file, the user can watch the video in real time via a data link connected to the server. We call this *video streaming*. Assuming the user plays the video at the same rate as it downloads the video, the simplest way of streaming from a server takes the following steps:

1. The user downloads a video frame, or *data chunk*, C, from the server.
2. The user plays the frame C.
3. Return to Step 1 and repeat.

This simple model would work well in reality if not for its parallel problem of scalability: if N users are trying to stream the same video from the same server, and if each user streams at a constant rate r, then the server's uplink requirement is approximately rN. Such a rapid rate of increase in upload bandwidth can add significant stress to the server as N becomes large.

For video content providers who do not have a budget to expand the physical uplink bandwidth, a natural question to ask is: Does there exist a way to serve a large number of peers who are trying to stream the same video, given a limited upload bandwidth from the server? The answer is affirmative, and one of the ways to efficiently achieve this goal is to use the technique called Peer-to-Peer (P2P) video streaming.

1.1 A Peer-to-Peer Streaming Network

The central idea for a Peer-to-Peer (P2P) video streaming system is to fully utilize *every user*'s upload bandwidth and storage space to help the server distribute the content. Instead of downloading only from the server, each user now is allowed to contact other peers and download parts of the file. Hence the total upload service requirement is now shared among the group of peers, rather than by the server alone.

We adopt the following model from [1]. Consider a network with M peers trying to stream a live video from a single server. For simplicity of analysis, assume the system progresses on a discrete time basis ($t = 1, 2, 3, ...$), video contents are distributed in the form of data packets called *chunks*, and the video is played at the rate of one chunk per time slot. At the beginning of each time slot, the server selects a peer uniformly at random from the network

and uploads the most recent chunk. Due to the limited upload bandwidth at the server, each peer employs an N-chunk buffer to cache the video stream, where location 1 denotes the most recent chunk that is currently being uploaded by the server, and location N is where the video is currently being played. During each time slot, every peer is allowed to contact another peer randomly selected from the rest of the network, which we call *target*, and download at most one chunk of content that is available in target's buffer locations 2 to $N-1$. At the end of the time slot, the chunk at buffer location N, if it exists, is played, and all other chunks in the buffer are shifted by one location towards the end of the buffer. An illustration of a peer buffer is given in Figure 1.2.

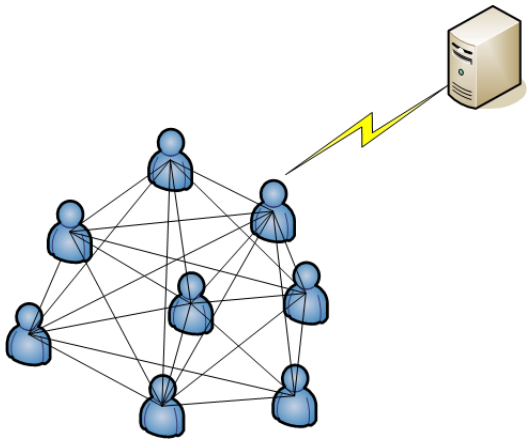


Figure 1.1: Structure of a P2P Streaming Network.

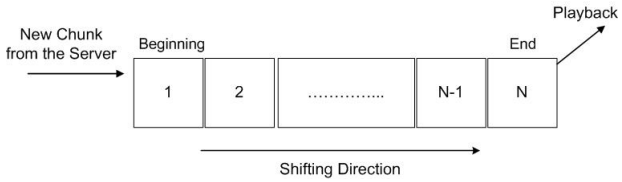


Figure 1.2: Illustration of A Peer's Buffer

1.2 Definitions

To facilitate the analysis in later sections, we define a set of notations and operators associated with the distribution process which will be formally defined in Section 1.3.

Definition 1. *Let the buffer length be N, the* ***buffer state*** *of peer i is defined to be the N-element vector $x_i \in \{0,1\}^N$, where $x_i(n) = 1$ indicates there is a chunk in buffer location n, and $x_i(n) = 0$ indicates buffer location n is empty. For example, when $N = 5$ and $x_i = 00001$, it means peer i has only one chunk, which is in buffer location 5.*

Definition 2. *The* ***buffer state space*** *is defined to be the set of all possible buffer states: $\mathcal{X} = \{0,1\}^N$.*

Definition 3. *A **deterministic downloading policy** is a function* $\delta(x_i, x_j) : \mathcal{X}^2 \to \{2, ..., N-1\}$. *Given a peer's buffer state* x_i, *and its target's buffer state* x_j, *the downloading policy* $\delta(x_i, x_j)$ *maps* (x_i, x_j) *to an index of the buffer location, in which the peer intends to download the chunk. Unless otherwise specified, we assume all peers use the same downloading policy.*

Definition 4. *A **shifting operator** S:* $\mathcal{X} \to \mathcal{X}$ *is defined such that if* $y = Sx,\ x, y \in \mathcal{X}$, *then* $y_{i+1} = x_i, \forall i \in \{1, ..., N-1\}$, *and* $y_1 = 0$.

Definition 5. *A **downloading operator*** $D_i(x) : \mathcal{X} \to \mathcal{X}, i \in \{1, ..., N\}$, *is defined such that if* $y = D_i(x),\ x, y \in \{0,\ 1\}^N$, *then* $y_j = x_j, \forall j \in \{1, ..., N\} \backslash \{i\}$, *and* $y_i = 1$.

1.3 The Content Distribution Process

To summarize, we now formally define the video distributing process by specifying the exact actions taken by the server and peers within one time slot.

1. Time slot $t = k$ begins.
2. Each peer plays the current chunk that is in buffer location N, if it exists.
3. The server selects a peer, indexed by l, uniformly at random from the network and uploads to peer l the most recent video chunk C_{new} into buffer location 1 of the peer.
4. For every peer i who was not selected by the server in the previous step ($i \in \{1, 2, ..., M\} \backslash \{l\}$):
   (a) The peer selects a target j uniformly at random from the network, with $j \neq i$.
   (b) The peer *downloads* a chunk C_d from the set of *downloadable chunks*, denoted by $\mathcal{D}(x_i, x_j)$, which is the set of chunks available at the target's buffer locations $2, 3, .., N-1$, but are not possessed by the downloading peer. The choice of d is made by the downloading policy δ, i.e. $d = \delta(x_i, x_j)$. If $\mathcal{D}(i, j)$ is empty, the peer does not download anything.
5. For all peers, the chunk in the N-th location of the buffer, if it exists, is played, and all other chunks in the buffer are shifted towards the end of the buffer by one buffer location.
6. Time slot $t = k$ ends. Set $t = k + 1$ and return to Step 1.

Note that only the chunks in buffer locations $2, 3, ..., N-1$ are downloadable, which means buffer location 1 is always *empty* for *all peers* at the beginning of a time slot. Also, since only one user was selected by the server in the previous iteration, only one peer in the network has a chunk in buffer location 2 at the beginning of any time slot.

Chaper 2

Probabilistic Models for Peer-to-Peer Live Streaming

In this chapter, we describe three probabilistic models that can be used to describe the dynamics of this P2P streaming network: the network state model, the user state model , and the marginal probability model. While the network state model describes the system with the highest degree of comprehensiveness compared to the other two, the relative simplicities of user state and marginal probability models provide a better ground for performing more in depth analysis on how the system behaves.

2.1 The Network State Model

The network state model is the most general of all three and describes the system most accurately. Due to its details, however, the network state model is very difficult to analyze to obtain performance-specific results. Nevertheless, it provides important insights on the nature of the dynamic system, as well as what assumptions could be made to simplify the model.

Definition 6. *Let the number of peers be M and the buffer length be N. A* ***network state*** *is a vector $\psi \in \{0,1\}^{MN}$, which encompasses the buffer states of all peers in the P2P network, defined by:*

$$\psi := [x_1\ x_2\ ...\ x_{M-1}\ x_M]\,, \tag{2.1}$$

where $x_i \in \{0,1\}^N, \forall i \in \{1,2,...,N\}$.

Let $\Psi(t)$ denote the network state at the beginning of time slot t, we observe that the random variables $(\Psi(t))$ form a *finite-state Markov process* $\{\Psi(t) : t \in N\}$. It is easy to identify that the transition probability p_{ψ_i,ψ_j} depends on the *deterministic downloading policy* δ employed by the network. Hence, we can express the conditional probability of the network states as:

$$P\left(\Psi(t+1) = \tilde{\psi}\right) = \sum_{\psi} p_{\psi\tilde{\psi}}(\delta) P\left(\Psi(t) = \psi\right). \tag{2.2}$$

If we know the expressions for the transition probabilities $p_{\psi_i,\psi_j}(\delta)$ for all $\psi_i, \psi_j \in \{0,1\}^{MN}$ given any downloading policy δ, equation (2.2) along with an initial condition $\Psi(1) = \psi(1)$ will allow us to fully describe the distribution of the system in any time slot t. Unfortunately,

finding an exact expression for $p_{\psi_i,\psi_j}(\delta)$ can easily be prohibitively computationally expensive. In a common P2P network with the network size $M \approx 1000$ and buffer size $N \approx 20$, finding $p_{\psi_i,\psi_j}(\delta)$ involves computing a probability mass function with $2^{1000*20*2} = 2^{40000}$ states, an astronomically large number. More complex tasks such as using numerical methods to find optimal downloading policies may simply be computationally infeasible.

While lacking computational tractability, the network state model provides a basis on which simplified versions can be built. The next two models can be treated as extensions of the general network state model, by adding additional independence assumptions. The user state model assumes independent and identical distributions for each peer's states, and the marginal probability model, in addition the assumption of inter-peer independence, assumes that the marginal chunk distributions at different buffer locations within a single peer are independent.

Note: (Uniqueness of Equilibrium Distribution) Based on the network model, we can see that for a given deterministic downloading policy the dynamics of the network is described by an *irreducible aperiodic finite-state Markov process*. The irreducibility comes from the fact that one can always find a sample path with positive probability that goes from one state to another. The aperiodic property can be shown as follows:

Suppose we start with a network state $\Psi_0 = \hat{\psi} = [x_1\ x_2\ ...\ x_i\ ...\ x_n]$ where $x_k = [0\ 1\ 1\ ...\ 1]$ if $k = i$ and $[0\ 0\ 0\ ...\ 0]$ otherwise. In the current time slot, peer i will be served by the server with probability $\frac{1}{M}$ to maintain its buffer state $x_k = [0\ 1\ 1\ ...\ 1]$, whereas all other peers will stay in the state $[0\ 0\ 0\ ...\ 0]$ with probability $(1-\frac{1}{M})^{M-1}$, which is the probability that none of the them contacts peer i in this time slot. The probability that the network will remain in state Ψ after one time slot is therefore $p_{\hat{\psi}\hat{\psi}}(1) = \frac{1}{M}(1-\frac{1}{M})^{M-1} > 0$. Hence state $\hat{\psi}$ has a period equal to 1. Since the Markov chain is irreducible, all states have the same period, and hence the process is aperiodic by definition.

Finally, as a well-known result in the literature of stochastic processes, it is shown in [3] that all irreducible aperiodic finite-state Markov processes have a unique equilibrium distribution.

2.2 The User State Model

While the network state model is able to fully describe the network, we are more interested in the performance factors experienced by each individual peer. The user state model focuses on the steady-state distribution of an average peer's buffer, by assuming that when the total number of peers M becomes very large, the state at each peer is essentially independent from one another. Hence, instead of trying to solve for the exact states for the entire system, the user state model simply states that when a peer contacts a target, the buffer state it observes at the target is equivalent to a random sample drawn from a certain distribution.

The user state model is essentially based on the heuristic in statistical physics called the

Mean-Field Theory, which simplifies the iteraction between any single peer and the rest of the group so that the peer is expected to keep sampling independently from the *mean* buffer state distribution of the rest of the peers. Therefore, we can assume that in stead-state, all peers have an identical and independent buffer state distribution, which equals to the expected distribution of the whole network, which is observed by a peer during every time slot. This reduces the total number of states from 2^{MN} to 2^N. The user state model is essentially the same as the Density Dependent Jump Markov Process model proposed in [2].

It is worth noting that the model is not completely accurate because in reality, every peer has a *unique* buffer state distribution at any times, but the model is assuming that all peers take on the mean buffer state distribution when being sampled as a target. But the error is generally diminishing as the number of peers $M \to \infty$.

Following the content distributing process described in Section 1.3, a steady-state buffer state distribution P_s is described by the a solution to the fixed-point equation below:

$$P_s(x) = \frac{1}{M} \sum_{x_i \in X_1} P_s(x_i) + \left(1 - \frac{1}{M}\right) \sum_{(x_i,x_j) \in X_2(\delta)} P_s(x_i) P_s(x_j), \forall x \in \mathcal{X}, \tag{2.3}$$

where X_1 is the set of all states that will become x after downloading the chunk in buffer position 1, or: $X_1 = \{x_i : SD_1(x_i) = x\}$.(In fact X_1 always has only two elements, differing by the N-th bit). $X_2(\delta)$ is the set of pairs of states (x_i, x_j) such that when a peer with state x_i downloads from a target with state x_j according to the downloading policy δ, the resulting state is x, or: $X_2(\delta) = \{(x_i, x_j) : SD_{\delta(x_i,x_j)}(x_i) = x\}$.

2.3 The Marginal Probability Model

Although the user state model significantly reduces the state space, computations are still challenging as the length of buffer N becomes large. To further simplify the problem, the marginal probability model, first proposed by Zhou *et al.* [1] makes the assumption that the states of individual buffer locations are independent from each other. Hence the joint distribution of a particular buffer state can be written in terms of the *product* of the marginal probabilities of all buffer locations, reducing the dimension of the state space from 2^N to N.

Assuming identical and independent peer state distribution for all peers, let p_i denote the steady-state marginal probability that buffer location i has a chunk in the current time slot:

$$p_i = \sum_{x \in \{x \in \mathcal{X} : x(i) = 1\}} P_s(x). \tag{2.4}$$

The marginal probabilities $\{p_i, i \in \{1, 2, ..., N\}\}$ are defined by the following equations [1]:

$$\begin{aligned} p_1 &= 0 \\ p_2 &= \tfrac{1}{M} \\ p_{i+1} &= p_i + p_i(1 - p_i)s_i(\delta), \forall i \in \{2, 3, ..., N-1\}, \end{aligned} \tag{2.5}$$

where $s_i(\delta)$ denotes the probability that one piece will be downloaded to the buffer position

i in a given time slot, given a deterministic downloading policy δ. In later sections, the term $s_i(\delta)$ will take on specific expressions as we focus on particular classes of downloading policies.

While it becomes more obvious in later sections, we can already see that the independence assumption on buffer states is not true in reality, because any deterministic downloading policy will introduce correlations among different locations in the same buffer. The simplicity of the marginal probability model and its relative accuracy, however, allows us to study many interesting performance factors qualitatively on large buffer sizes, such as the impact of various downloading policies on playback continuity and the uniqueness of distribution of buffer states.

Chaper 3

Studies Using the Marginal Probability Model

In this chapter, we employ the Marginal-Probability (MP) model to study a collection of topics that concern the performance of a peer-to-peer video streaming network. We begin by investigating the playback-continuity-maximizing downloading policies.

3.1 Optimal Permutation-based Chunk Selection Heuristic

One of the most important performance measures for a P2P video streaming network is the playback continuity experienced by the peers. In our context, it is defined by the probability of the peer being able to "play" a frame at each time slot. If such a probability is low, the viewing experience is more likely to be degraded due to skipped frames or noticeable playback interruptions. Relating to the network model in Section 1.3, this quantity is represented by the marginal probability of seeing a chunk in buffer location N:

$$\text{Playback Continuity} := p_N = \sum_{x \in \{x \in \mathcal{X} : x(N)=1\}} P(x), \tag{3.1}$$

3.1.1 Permutation-based Chunk Selection Policies

It can be observed from the distributing process described in Section 1.3 that the only controllable input to the system is the downloading policy δ. Hence our goal will be to find a policy so that the steady-state marginal probability p_N is maximized. Generally, as defined in Section 1.2, a downloading policy is a function that depends on the state of both the peer itself *and* the state of the target. To simplify the case and obtain more intuitive understanding of the system, here we restrict our study to policies that be described by a set of deterministic *preference permutation*, defined as below.

Definition 7. *A **preference permutation** π is a permutation of the set of buffer location indices $\{2, 3, ..., N-1\}$. A downloading policy δ_π based on a deterministic preference permutation π is to always download the piece whose index is closest to the beginning of the permutation, i.e.*

$$\delta_\pi(i,j) = \min_m \{m \in \{2, 3, ..., N-1\} : \pi(m) \in \mathcal{D}(i,j)\}. \tag{3.2}$$

Below are some typical preference permutations, discussed also in [1].

- *Earliest Deadline First (EDF)*: $\pi = \{N-1, N-2, ..., 3, 2\}$
- *Latest Deadline First (LDF)*: $\pi = \{2, 3, ..., N-2, N-1\}$
- *Mixed*: $\pi = \{2, 3, ..., k-1, N-2, N-1, ..., k+1, k\}$ for some threshold k, $k \in \{3, 4, ..., N-2\}$

Figure 3.1 gives a comparison of performance among difference choices of permutations. It has been shown experimentally in [1] that by varying the threshold k, one could achieve a better playback using Mixed permutations compared to LDF and EDF. Roughly speaking, the EDF policy chooses piece by its urgency to playback, and LDF prefers chunks that tend to be more rare in the network (since the marginal probabilities p_i's are strictly increasing in i), and the Mixed policy essentially represents a compromise between the two.

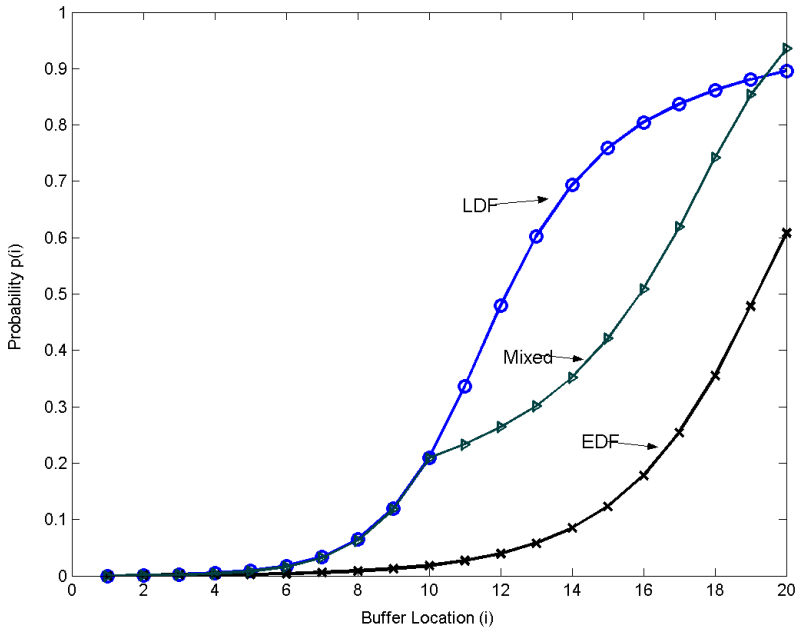


Figure 3.1: Steady-state Buffer Marginal Probabilities due to Different Permutations

Given a buffer length N, the total number of unique permutations of $\{2, 3, ..., N-1\}$ is $(N-2)!$. Such a size of posible permutations increases too rapidly in N for an exhaustive search method to tackle the problem of finding the optimal permutation among all possible permutations. We therefore propose a computationally efficient permutation search heuristic in the following section in an attempt to find near optimal permutations.

3.1.2 The Hardness-First Heuristic

In some sense, the definition of a permutation-based strategy assigns a ranking of *importance* between the buffers, since the policy requires to download the piece that is closest to the head of the permutation. In EDF, chunks that have a smaller time to playback are considered more important, whereas in LDF, higher importances are given to chunks that are rarer in

the network. In the Hardness-First heuristic, we judge the importance by the probability of a chunk appearing in the set of downloadable pieces, denoted by the term *hardness*, defined as below:

Definition 8. *The **hardness**, h_i, of the chunk in buffer location i is defined by $h_i = 1 - p_i(1 - p_i)$.*

Interpretation: It is assumed that in steady-state all peers have the same buffer-state distribution, hence the same marginal probability for each buffer location. Therefore, the term $p_i(1 - p_i)$ can be interpreted as the joint probability that the target possesses the chunk in buffer location i (p_i), while the peer does not ($1 - p_i$). In another word, a high $p_i(1 - p_i)$ means chunk i is *"easier"* to be seen in the pieces that can be downloaded after contacting a random target, and hence a higher $1 - p_i(1 - p_i)$ suggests chunk i is *"harder"* to be among the set of downloadable pieces.

Based on this interpretation, we would like to find a permutation such that the chunks are ranked in a decreasing order of hardness, as is described in the following definition:

Definition 9. *A **hardness-first permutation** is a preference permutation, π_h, whose steady-state marginal distribution satisfies the following ordering of hardness:*

$$h_{\pi(i)} \geq h_{\pi(j)}, \forall i \geq j, i, j \in \{2, 3, ..., N-1\}. \tag{3.3}$$

3.1.3 The Algorithm for Finding Hardness-First Permutations

We employ a greedy algorithm to converge to a hardness-first permutation, starting from any arbitrary permutation of $\{2, 3, ..., N-1\}$. In each iteration the algorithm computes the stead-state marginal chunk distribution and updates the permutation sequence to be *more hardness-oriented* based on the current marginal distribution:

- Phase 1: Calculate the buffer's steady-state marginal probabilities $\{p_i\}_k$ given a preference permutation π_k.
- Phase 2: Update π_k to π_{k+1} based on $\{p_i\}_k$.

The general structure of the algorithm is shown in Figure 3.2.

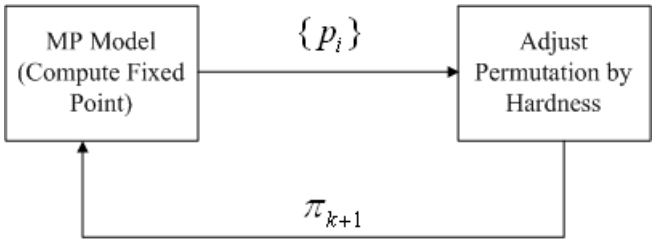


Figure 3.2: Illustration of the Greedy Search for a Hardness-First Permutation

Calculating Steady-State Buffer Distribution: The first phase calculates the steady-state marginal distribution of the buffer locations given a permutation π, by finding the

fixed-point solution to (2.5). Let i be the m-th element in π, the probability of a peer choosing to download chunk i (s_i in (2.5)) can be expressed as:

$$s_i = (1 - \frac{1}{M}) \prod_{\forall j<m, j\in\{2,...,N-1\}} [1 - p_{\pi_i}(1 - p_{\pi_i})] \tag{3.4}$$

Standard numerical techniques can then be used to solve for the solutions of the fixed-point equations defined by (2.5) and (3.4).

Updating permutation π_k based on $\{p_i\}_k$: The goal of the second phase is to come up with an updated permutation that is in some sense more hardness-oriented. A naive solution will be to calculate the hardness h_i for all i using $\{p_i\}_k$, and produce the updated permutation by putting the chunk indices in a nonincreasing ordering of the h_i's. This approach however, does not converge in many cases. The convergence issue is mitigated by taking steps of smaller changes. In our implementation, the update step simply *swaps* two adjacent elements with the smallest decreasing (or largest increasing) in hardness, i.e:

$$\pi_{k+1} = (\pi_k(1), \pi_k(1), ..., \pi_k(i_{swap}+1), \pi_k(i_{swap}), \pi_k(i_{swap}+2), ..., \pi_k(N-1)), \tag{3.5}$$

where

$$i = \arg \min_{2\le i\le N-2} (\pi_k(i) - \pi_k(i+1)). \tag{3.6}$$

However, convergence is not guaranteed in general.

3.1.4 Numerical Calculation and Simulation Results

Converging to a Hardness-First Permutation: Figure 3.3 illustrates the result of running the algorithm in Section 3.1.3. The first subgraph compares the marginal distribution that is resulted by using the initial permutation (which is set to EDF in this case) and the final permutation, both calculated numerically by the MP model. The second subgraph plots the hardness (h_i) for each buffer location, from which it can be observed that the h_i's converge to values with a nonincreasing ordering in the final permutation. The last subgraphs illustrates the structure of the final permutation, where the permutation starts with a LDF up to buffer location 8 and then "jumps" between the beginning and the end of the rest of the buffer locations.

Performance: In Figure 3.4(a) and Figure 3.4(b), steady-state marginal distributions using different permutations are compared. The Hardness-First permutation outperforms all of LDF, EDF and Mixed in terms of playback continuity (p_N), although the advantage compared to Mixed is very small.

Since both LDF and EDF belong to the class of Mixed permutations, we proceed to compare the performance of Hardness-First permutations against the best Mixed permutation (obtained by optimizing over all position threshold k) for a given buffer size. Define a comparison metric, the *Miss Rate Ratio*, as the ratio of the probability of not seeing a chunk

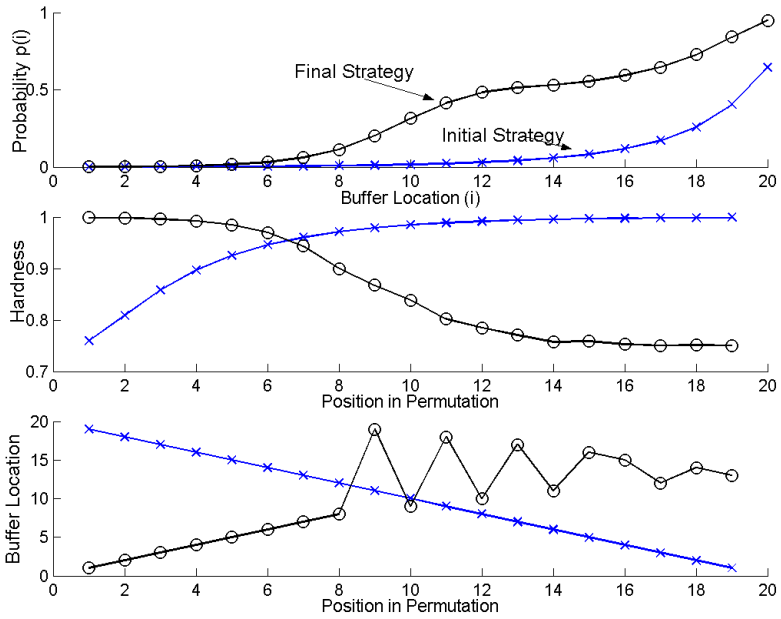
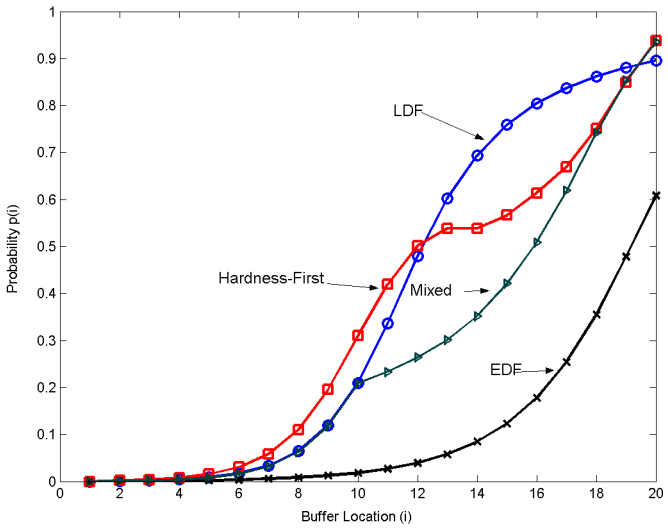


Figure 3.3: Converging to a Hardness-First Permutation

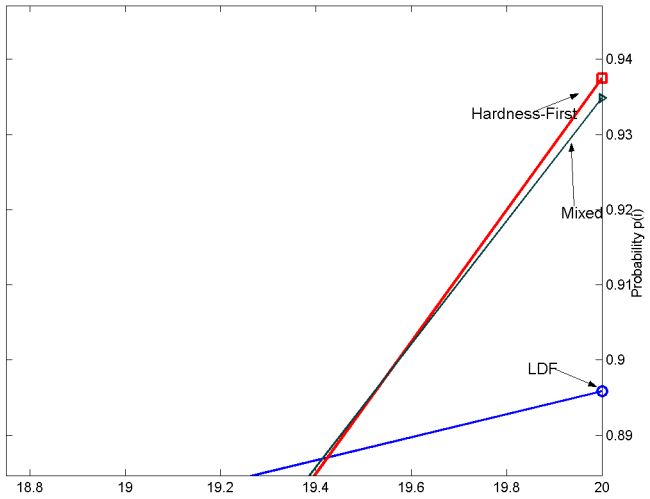
at buffer location N between the hardness-first and optimal mixed permutations, i.e.:

$$\text{Miss Rate Ratio} = \frac{1 - p_N(\pi_{hardness-first})}{1 - p_N(\pi_{optimal-mixed})}. \tag{3.7}$$

With the number of peers $M = 1000$, Figure 3.5 shows the comparison in playback continuities and the Miss Rate Ratio computed by the MP-model. The hardness-first permutation appears to outperform the best Mixed permutation in the buffer size range of $15 \leq N \leq 39$, since the Miss Rate Ratio is always less than 1. However, such advantage is quite small in an absolute scale. Also, the plot suggests that by increasing the buffer size by about 2, the continuity of optimal mixed permutation can match up with that of the hardness-first permutation with the original buffer size.



(a) Comparison of Performances among Hardness-First, Mixed, EDF and LDF (Simulations)



(b) Zoomed-in Version of Figure 3.4(a)

Figure 3.4:

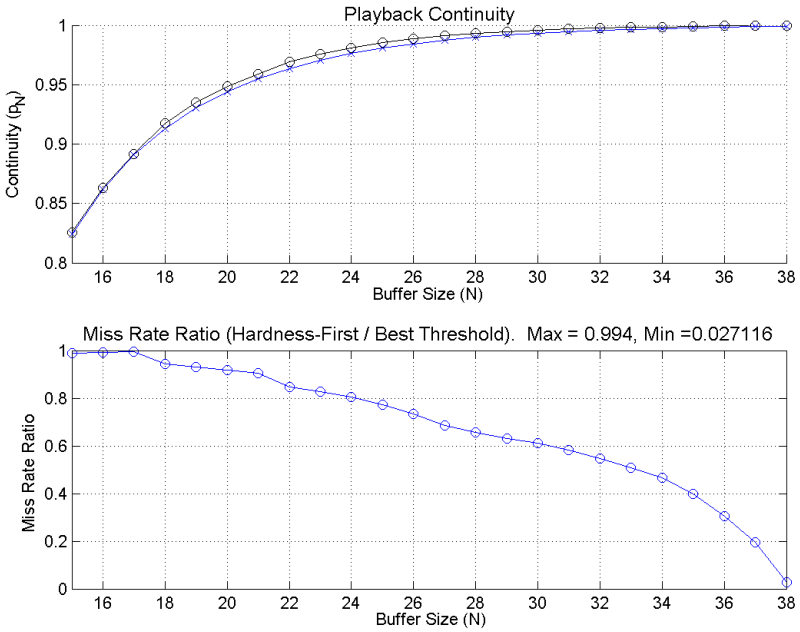


Figure 3.5: Continuity Comparison between Hardness-First and the Best Mixed Policy

3.2 Uniqueness of the Equilibrium Marginal Distribution

Besides searching for a continuity-maximizing permutation, we are also concerned about the stability of the system, namely, that given a specific permutation sequence whether there exists a unique marginal distribution. It is shown in Section 2.1 that the network possesses a unique equilibrium distribution in terms of the network states. However, the uniqueness of the equilibrium only guarantees the convergence in time to an equilibrium joint distribution, and implies little of what a peer would expect to experience in terms of the playback continuity.

A simple example: Consider a simple Markov chain with two states: *good* and *bad*, for which the transition diagram is drawn in Figure 3.6. Observe that this Markov process is aperiodic and irreducible and hence has a unique equilibrium distribution. However a observer may witness drastically different experience (good or bad) depending on if the network starts in a good state or not, given that the transition probability between the two aggregate states are extremely small.

While by numerically solving for the fixed-point solutions of (2.5) and (3.4) we are able to obtain *a* steady-state marginal distribution, it is not immediately clear whether such a solution is unique. The property that we are interested in finding out is whether there are multiple solutions of the marginal probability distributions that satisfy the fixed-point equations.

We show in this section that the fixed-point marginal distribution is indeed unique for the LDF, EDF and Mixed policies. However, similar attempts for obtaining uniqueness results

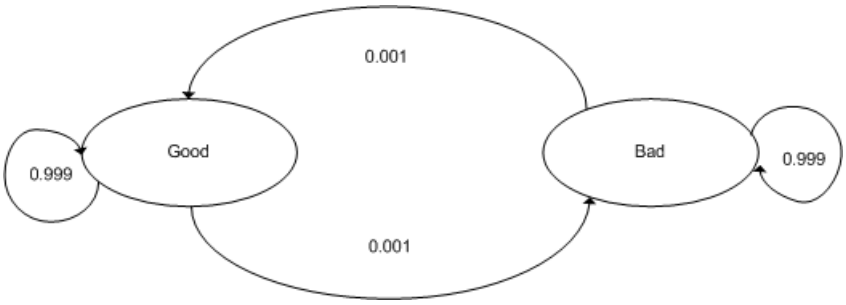


Figure 3.6: Transition Diagram for the Simple Markov Chain with Two States

for more general permutations remain unsolved.

Proposition 10. *Based on the MP model, the steady-state marginal distributions* $\{p_i\}$ *by using LDF, EDF and Mixed with any threshold are all unique.*

Proof. **Uniqueness of LDF**: It can be shown [1] that when LDF is used the s_i takes on a simple form of:

$$s_i = 1 - p_i. \tag{3.8}$$

Thus (2.5) reduces to:

$$\begin{aligned} p_1 &= 0 \\ p_2 &= \tfrac{1}{M} \\ p_{i+1} &= p_i + p_i(1-p_i)^2, \forall i \in \{2, 3, ..., N-1\}. \end{aligned} \tag{3.9}$$

Note that p_{i+1} is a function of p_i for all $i \in \{1, 2, .., N-1\}$. Hence given the initial conditions $p_1 = 0, p_2 = 1$, p_i is uniquely determined for all $i \in \{1, 2, .., N\}$.

Uniqueness of EDF/Mixed: Given the uniqueness result for LDF, it suffices to show uniqueness only for EDF, since Mixed can be treated as an EDF permutation with buffer length being $N - k + 1$ and with p_2 replaced by p_k. When EDF is, the probability of s_i can be expressed as: [1]

$$s_i = 1 - p_N + p_{i+1} \tag{3.10}$$

Hence the fixed-point equation for $p_i, k+1 \leq i \leq N-1$ becomes:

$$\begin{aligned} p_{i+1} &= p_i + p_i(1-p_i)(1-p_N+p_{i+1}) \\ &= p_i + p_i(1-p_i)(1-p_k-p_N+p_i), \forall i \in \{k, k+1, ..., N-2\}, \end{aligned} \tag{3.11}$$

with the boundary condition for the N'th buffer location

$$p_N = p_{N-1} + p_{N-1}(1-p_{N-1})(1-p_k-p_N+p_{N-1}). \tag{3.12}$$

The form of p_{i+1} suggests that it is a function of p_i and p_N. Observe that since p_k is a constant value determined by the LDF equations, p_{k+1} becomes a function of P_N only. Using an induction argument, we can easily verify that each $p_{i+1}, i \geq k+1$ is in fact a function of P_N only.

To prove uniqueness of fixed-point for (3.11) and (3.12), we claim that p_{i+1} decreases as P_N increases for all $i \in \{k, k+1, ..., N-1\}$,i.e.:

$$\frac{dp_{i+1}}{dp_N} < 0, \forall i \in \{k, k+1, ..., N-2\}. \tag{3.13}$$

Therefore, given one value of $p_N = \hat{p_N}$ that satisfies the fixed-point equations, if we were to move p_N away from $\hat{p_N}$, (3.12) will *never* be satisfied with another value $p_N = \tilde{p_N} \neq \hat{p_N}$.

Using the Chain Rule of calculus, $\frac{dp_{i+1}}{dp_N}$ can be expressed as:

$$\begin{aligned} \frac{dp_{i+1}}{dp_N} &= \frac{\partial p_{i+1}}{\partial p_i}\frac{dp_i}{dp_N} + \frac{\partial p_{i+1}}{\partial p_N}\frac{dp_N}{dp_N} \\ &= \frac{\partial p_{i+1}}{\partial p_i}\frac{dp_i}{dp_N} + \frac{\partial p_{i+1}}{\partial p_N}. \end{aligned} \tag{3.14}$$

From (3.11), we have:

$$p_{i+1} = \frac{p_i + p_i(1-p_i)(1-p_k+p_N)}{1-p_i(1-p_i)}, \forall i \in \{k, k+1, ..., N-2\}. \tag{3.15}$$

Taking the partial derivative of p_{i+1} with respect to p_N, we get:

$$\frac{\partial p_{i+1}}{\partial p_N} = -\frac{p_i(1-p_i)}{1-p_i(1-p_i)} < 0. \tag{3.16}$$

The inequality in (3.16) is due to the fact that $p_i \in (0,1)$.

Let $\sigma = 1 - p_k - p_N$. Taking the partial derivative of p_{i+1} with respect to p_i, we obtain:

$$\begin{aligned} \frac{\partial p_{i+1}}{\partial p_i} &= \frac{[1+\sigma(1-2p_i)]\,[1-p_i(1-p_i)] - (2p_i-1)\,[p_i+\sigma p_i(1-p_i)]}{[1-p_i(1-p_i]^2} \\ &= \frac{1}{A}\,[1-p_i(1-p_i)+\sigma(1-2p_i)-\sigma p_i(1-p_i)(1-2p_i)+p_i(1-2p_i)+\sigma p_i(1-p_i)(1-2p_i)] \\ &= \frac{1}{A}\,[1-p_i(1-p_i)+(1-2p_i)(\sigma+p_i)] \\ &= \frac{1}{A}\,\{1-[\sigma(1-p_i)+(\sigma+p_i)p_i]\}\,, \end{aligned} \tag{3.17}$$

where $\frac{1}{A} = \frac{1}{[1-p_i(1-p_i)]^2}$.

Due to the relationships $0 < p_k \leq p_i \leq p_N < 1$, we have:

$$\sigma = 1 - p_k - p_N \Longrightarrow \sigma \in (-1, 1) \Longrightarrow |\sigma| < 1$$
$$\sigma + p_i = 1 - p_k - (p_N - p_i) = 1 + (p_i - p_k) - p_N \Longrightarrow (\sigma + p_i) \in (0, 1) \Longrightarrow |\sigma + p_i| < 1$$

Since the term $\sigma(1 - p_i) + (\sigma + p_i)p_i$ can be viewed as a linear combination of σ and $\sigma + p_i$, we have:

$$\frac{\partial p_{i+1}}{\partial p_i} > 0, \forall i \in \{k, k+1, ..., N-1\}. \tag{3.18}$$

Induction: When $i = k$, p_k is a constant. Hence (3.14) reduces to

$$\frac{dp_{k+1}}{dp_N} = \frac{\partial p_{k+1}}{\partial p_k} \cdot 0 + \frac{\partial p_{k+1}}{\partial p_N} = \frac{\partial p_{k+1}}{\partial p_N} < 0 \tag{3.19}$$

Suppose $\frac{dp_m}{dp_N} < 0$ for some $m \in \{k+2, k+3, ..., N-1\}$, from (3.16) and (3.18) we have:

$$\frac{dp_{m+1}}{dp_N} = \frac{\partial p_{m+1}}{\partial p_m}\frac{dp_m}{dp_N} + \frac{\partial p_{m+1}}{\partial p_N} < 0. \tag{3.20}$$

Finally, denote the value of the right-hand-side of (3.12) by $\tilde{p_N}$, we get:

$$\frac{dp_{\tilde{p_N}}}{dp_N} < 0 \tag{3.21}$$

Hence $p_N = \tilde{p_N}$ has only one solution of p_N. Since every $p_i, i > k$ is a function of p_N as shown above, this means there exists a unique set of p_i's that satisfies (3.12) and (3.11).

□

3.3 Stability Implications of Incentive Strategies

While the P2P streaming model considered in this paper assumes that each peer behaves in a cooperatively manner, in the sense that whenever a target is contacted by a peer, the target will always truthfully report its current collection of chunks and upload according to the downloader's requests. In reality, however, a selfish peer will prefer *not* to upload, because uploading consumes its bandwidth resources and does not bring any direct benefits. Under the original network setup, a selfish peer can do so by, for example, sending out false messages and pretending that it does not have any chunk. To prevent such peers from refusing to upload, incentive strategies can be imposed so as to create motivations for cooperation.

Consider a typical incentive strategy called *tit-for-tat*, defined as follows:

Definition 11. *The **tit-for-tat constraint**: In each round, a peer can download a chunk from the target only if it also uploads a chunk to the target simultaneously.*

Under the tit-for-tat constraint, every peer, except for the one who downloads directly from the server, has its download amount enforced to be equal to its upload amount in each

iteration. The benefit in refusing to upload is therefore eliminated because doing so will proportionally hurt the peer's own downloading performance.

A natural question to ask is by how much the enforcement of the tit-for-tat will impact the performance of the network. Intuitively, we expect the steady-state continuity to be lower than the original situation due to the additional tit-for-tat constraint. Interestingly, it turns out that imposing the tit-for-tat strategy not only degrades the overall playback continuity of the network, which is expected, but also creates a *bistability* in the solutions to the fixed point of the marginal probability equations, where in one solution the continuity is every close to that without tit-for-tat (slight degradation), but in the other the continuity is severely degraded to close to $\frac{1}{M}$. In the following sections we first present an analytical solution that solves for the steady-state continuity with the tit-for-tat constraint based on the MP model, where the bistability phenomenon was first observed. We then show simulation results to empirically verify our analysis.

3.3.1 Modified Content Distribution Process with Tit-for-Tat

We now modify the content distribution process described in Section 1.3 to incorporate the tit-for-tat incentive strategy. We assume here that the number of peers M is even.

1. Time slot $t = k$ begin.
2. The set of peers $\{1, 2, ..., M\}$ is partitioned uniformly at random into a set of $M/2$ *pairs* $\mathcal{P} = \{(a_i, b_i)\}_{i=1}^{M/2}$ such that every peer belongs to one and only one pair. We call the two peers in one pair *partners*.
3. The server selects a peer, indexed by l, uniformly at random from the network and uploads to l the most recent video chunk C_{new}.
4. Peer l's partner m downloads from l according to the downloading policy δ if and only if m possesses at least one chunk that l could have downloaded, had l not been chosen by the server. Note that this downloading action is unilateral as m does not actually upload to l even it could.
5. For every other pair $(a, b) \in \{(a_i, b_i) \in \mathcal{P} : l \notin (a_i, b_i)\}$,
   (a) If both a and b has at least one chunk that its partner can download, a and b downloads from each other using the downloading policy δ.
   (b) Otherwise, neither a or b downloads anything in this time slot.
6. For all peers, the chunk in the N-th location of the buffer, if it exists, is discarded, and all other chunks in the buffer are shifted towards the end of the buffer by one buffer location.
7. Time slot $t = k$ ends. Set $t = k + 1$ and return to Step 1.

3.3.2 Bi-stability Results using the Modified Marginal Probability Model

We modify the MP fixed-point (3.9) to capture the effect of tit-for-tat.

Definition 12. *The **probability of offering**, d, is the probability that during one contact a peer's collection of chunks is NOT a subset of its partner's collection in steady state. In other words, it is the probability that a peers has some chunks which its partner does not have.*

Due to the tit-for-tat requirement, a pair of partners can download from each other if and only if they are both able to offer each other some chunks. This event has probability d^2 in steady state, according to the definition above. We arrive at a modified fixed-point equations for the MP model with LDF as in (3.22). The only difference between this and the original MP model (in (2.5)) is the d^2 multiplied to the probability of download for a specific buffer location. Note that although the probability of offering does not apply to the single pair that includes the peer selected by the server; however in a large network such error is very small.

$$\begin{aligned} p_1 &= 0 \\ p_2 &= \tfrac{1}{M} \\ p_{i+1} &= p_i + p_i(1-p_i)s_i d^2, \forall i \in \{2,3,...,N-1\}. \end{aligned} \tag{3.22}$$

Main Result: When tit-for-tat is imposed in a network using LDF, there are two continuity solutions p_{N1} and p_{N2} to the steady-state marginal distribution fixed-point equations, with p_{N1} close to the original continuity p_N and p_{N2} close to $\frac{1}{M}$.

Analytical Derivation using MP Model: The main idea is to find two equations that relate the continuity p_N to the probability of offering d. The uniqueness will hence depend on the number of solutions for p_N and d.

Based on the fact that the downloading policy is LDF, from (3.4) we have the expression for s_{i+1} as:

$$s_{i+1} = s_{i+1}\left[1 - p_i(1-p_i)\right] \tag{3.23}$$

From (3.22) and (3.23), we get:

$$\sum_{l=3}^{i} p_l - p_{l-1} = -d^2 \sum_{l=3}^{i} s_l - s_{l-1},\ \forall i \in \{3,4,..,N\}, \tag{3.24}$$

which is equivalent to:

$$p_i - p_2 = -d^2(s_i - s_2),\ \forall i \in \{3,4,..,N\}, \tag{3.25}$$

Using (3.25) to substitute s_i in equation (3.22), we get:

$$p_{i+1} = p_i + p_i(1-p_i)\left[\frac{1}{M} - p_i + d^2(1-\frac{1}{M})\right], \forall i \in \{2,3,..,N-1\}, \tag{3.26}$$

Let $i = N - 1$ in equation (3.26). Since each p_i depends on only the values of $p_j, j \leq i$, we can compute the value of p_N iteratively given the values of d and p_2 (constant). This constitutes the first equation we need.

In order to find the second equation to solve for p_N and d, observe that the continuity p_N is equivalent to the expected number of chunks downloaded by a peer in one time slot. Since we also know that if both peers in a pair are able to offer its partner some chunk, each peer will download at least one chunk. Therefore the expected download rate, assuming neither of the peers in a pair is chosen by the server, becomes d^2. This observation leads us to the relationship:

$$p_N = \frac{1}{M} + (1 - \frac{1}{M})d^2 \tag{3.27}$$

Combining (3.26) and (3.27), the solutions for p_N and d are given by:

$$p_N = p_{N-1} + p_{N-1}(1 - p_{N-1}) \left[\frac{1}{M} - p_{N-1} + d^2(1 - \frac{1}{M}) \right] \tag{3.28}$$

$$p_N = \frac{1}{M} + (1 - \frac{1}{M})d^2 \tag{3.29}$$

(3.28) and (3.29) can be solved numerically. With $N = 40$ and $M = 1000$, Figure 3.7 shows the values of p_N calculated by the two equations, where the solutions are represented by the coordinates of where the two curves cross. As can be observed from the graph, the first solution appears to be where both d and p_N are close to zero, whereas the second solution occurs at a continuity of approximately 0.95, which is slightly lower than the 0.967 continuity computed by the MP model *without* tit-for-tat. Numerically, the two solutions are:

$$\text{Solution 1: } p_N = 0.9526, d = 0.9760 \tag{3.30}$$

$$\text{Solution 2: } p_N = 0.0001, d = 0 \tag{3.31}$$

Interpretation: *Solution 1* suggests that when most peers in the network have a rich collection of chunks, the probability that both partners have something to offer is high. Hence the additional tit-for-tat constraint has very little negative impact on the performance of the network. *Solution 2* can be verified by setting d to 0 in (3.28) and (3.29), which gives $p_N = \frac{1}{M}$. This reflects the intuition that when all of the peers have very little in their chunk collection to start with, the probability of offering is naturally very small. Hence the tit-for-tat constraint becomes very harsh in this situation and prevents the very few chunks from propagating throughout the network, and the playback continuity falls back to about $\frac{1}{M}$, meaning no peer-to-peer downloading is happening at all.

Simulations: We use simulation results to verify the predictions given by (3.30) and (3.31). In Figure 3.8(a), the tit-for-tat constraint is imposed *after* the network has reached steady-state using LDF without the tit-for-tat. The continuity appears to be very close to the .9526 predicted value. In Figure 3.8(b), however, the steady-state continuity is only about 0.0045 when we impose the tit-for-tat constraint from the beginning where every peer starts

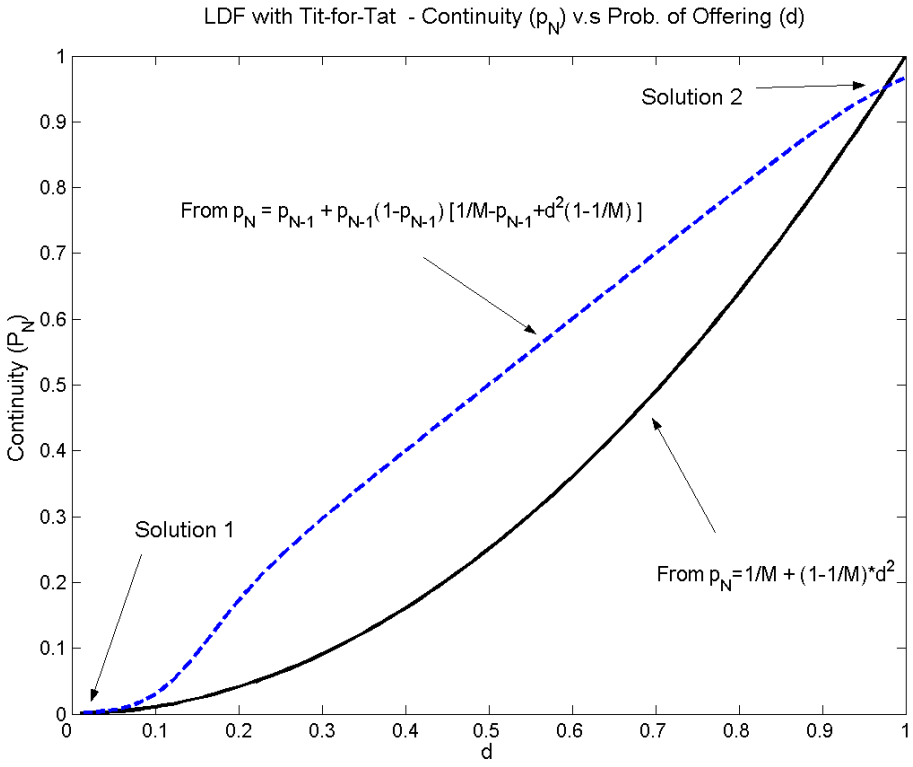
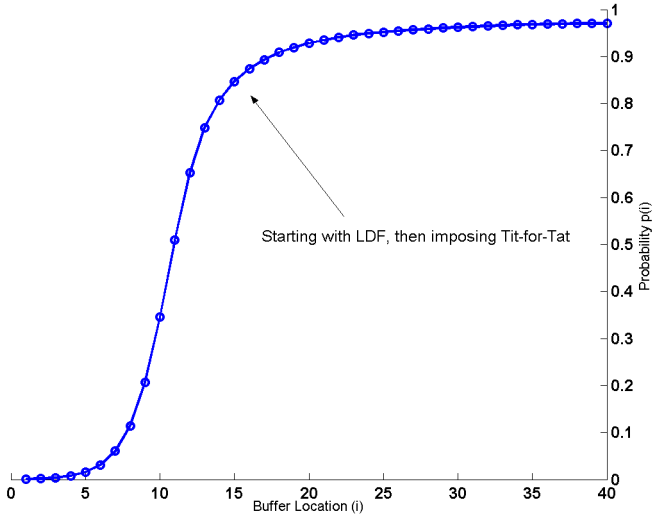


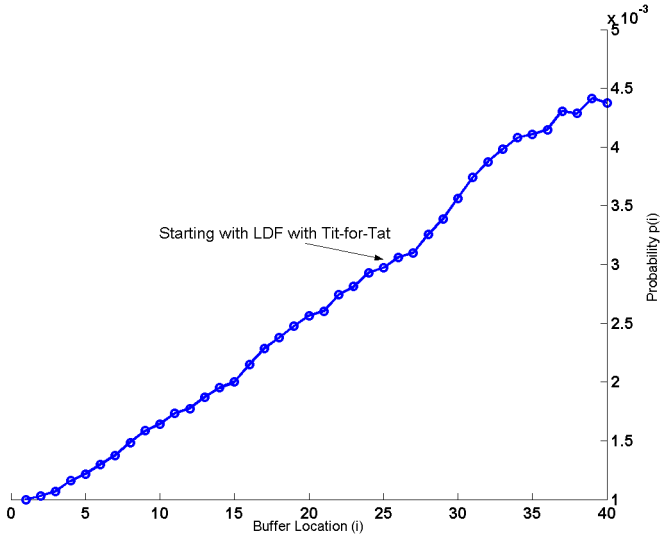
Figure 3.7: Solutions for p_N and d

with an empty buffer.

Note that the simulation results tend to be higher than the predictions. In the case of *Solution 2*, in particular, it is obvious that the probability of offering is strictly greater than 0 (although still very small) in reality, since there will be a few peers in the network who possess distinct chunks from the server, and the possible downloading among these peers will result in a positive d. We conjecture that this discrepancy is due to the fact that the dependency between chunk locations is not taken into account in the MP model.



(a) Simulations with tit-for-tat - Marginal Distribution 1



(b) Simulations with tit-for-tat - Marginal Distribution 1

Figure 3.8: Bistability Phenomenon in Simulations

Chaper 4

Studies using the User State model

While the marginal probability model provides a simple and intuitive way to analyze the system performance, its intrinsic inaccuracy, mainly due to the assumption that all buffer locations are independent, makes it difficult to draw more precise and rigorous conclusions on the behavior of the network. In this chapter, we turn our attention to the user state model. Instead of assuming a product-form joint distribution for the buffer as in the MP model, each *buffer state* in the peer-state model is treated individually, as was described in Section 2.2. This increased degree of granularity allows us to introduce a more refined definition of a peer's utility, as well as using dynamic programing to obtain insights in terms of optimal downloading policies.

It is worth noting that in this chapter we will pose the question of optimal downloading policy in a different angle: instead of asking for the optimal policy to maximize continuity for the *entire network*, which was the focus for Section 3.1, we would like to know, assuming a peer has *complete information* about all other peers' strategy, or equivalently, the equilibrium buffer-state distribution at the target, what downloading policy would it choose to maximize its *own* continuity, with the assumption that the peer size is reasonably large so that a selfish peer's local "cheating" behavior has no influence on the rest of the network. As we will see in later sections, this approach is partially motivated by the fact that given a known target distribution, the optimal strategy can be studied naturally using a dynamic programing formulation.

Note that the content distributing process is largely the same as before for a selfish peer. However, for the simplicity of analysis, we now assume this peer is not contacted by the server, and can download from buffer locations $\{1, 2, ..., N-1\}$ at the target.

4.1 The Buffer-State-Based Cost-to-Go

Consider a network where each peer maintains a buffer with a length N. We would like to have a quantity that relates a peer's current collection of chunks and its downloading policy with the expected utility in the future. According to the content distribution process described in Section 1.3, we say that a peer is *penalized* in a time slot, if the Nth buffer location is empty, which means the peer will have no frame to play for the current time slot. We therefore denote such an event (i.e. not having chunk N when it needs to be played), as charging the peer *1 unit of cost*. Using the this concept of cost, we define the peer's expected total costs in the next k time slots as the peer's *k-step cost-to-go*, a concept often seen in the literature of dynamic programming. [4]

Definition 13. *A peer's k-**step cost-to-go**, $C^k(x,\delta)$, is a function of the initial buffer state at time t, denoted by x^t, and downloading policy δ, defined as:*

$$C^k(x^t,\delta) = I\{x^t(N)=0\} + E\left[\sum_{m=t+1}^{t+k} \beta^{m-t} I\{x^m(N)=0\}\middle|\delta\right] \tag{4.1}$$

where $I\{\cdot\}$ is the identify function on a boolean variable such that $I\{A\}=1$ if and only if A is true, and β is a discount factor, $0 \le \beta \le 1$.

Observe that,

$$\lim_{k\to+\infty,\beta\to 1} \frac{1}{k} C^k(x^t,\delta) = p_N \tag{4.2}$$

where is p_N is the continuity measured used in previous sections.

Now, suppose that all other peers in the network are identical, such that in steady state a peer observes an iid sample of chunk collection drawn from a *target distribution* $\mathcal{F}$ every time it contacts a target. We proceed to give a recursive definition of a *minimum cost-to-go* and the associated *optimal downloading policy* which achieves the minimum cost.

Definition 14. *A peer's k-step **minimum cost-to-go** with an initial state x, $V^k(x)$ is defined by:*

$$\begin{aligned} V^0(x) &= 0 \\ V^k(x) &= I\{x^t(N)=0\} + \beta \min_\delta \textstyle\sum_{y\in\mathcal{X}} p_{xy}(\delta) V^{k-1}(y) \forall k \ge 1 \end{aligned} \tag{4.3}$$

where $p_{xy}(\delta)$ is the transition probability from state x to y using policy δ.

*A k-step **optimal downloading policy** $\hat{\delta}^k$, is defined by:*

$$\hat{\delta}^k = \arg\min_\delta \sum_{y\in\mathcal{X}} p_{xy}(\delta) V^{k-1}(y), \forall k \ge 1 \tag{4.4}$$

Note that, when $\beta < 1$ in the limiting case $C^k(x,\delta) \to C(x,\delta)$,$V^k(x) \to V(x)$, and $\hat{\delta}^k \to \hat{\delta}$ as $k \to \infty$.

4.2 A Necessary Condition for Optimal Downloading Policy

We present in this section a simple result that states given *any* buffer-state distribution at the target, *all* optimal downloading policies, as defined in (14), must download the chunk which is going to be played in the next time slot, if such a download is possible. This means any downloading policy that does not do so (e.g. the LDF permutation introduced earlier), is not a robust downloading policy in the sense that in such a network a selfish peer can benefit by switching to a better policy.

Denote the sequence of the set of chunks that can be downloaded (both available at the target and needed by the peer) in each future time slot by $(D_n), n = t, t+1, ...$ Let us

define the cost-to-go given a sample path of target collections (D_n). Note that in this case the cost-to-go no longer contains an expectation over the target distribution, but becomes a deterministic function of x^t, δ and (D_n).

Definition 15. *A peer's cost-to-go given a sequence of the future downloadable sets $(D_n), n = t, t+1, ...$ is:*

$$C^k\left(x^t, \delta, (D_n)\right) = I\{x^t(N) = 0\} + \sum_{m=t+1}^{t+k} \beta^{m-t} I\{x^m(N) = 0\} \tag{4.5}$$

where $x^m(N)$ now is function of x^t, δ and (D_n), $\forall m \geq t+1$.

Similarly, the minimum cost-to-go given (D_N),$V^k(x, (D_N))$, is defined by:

$$\begin{aligned} V^0(x, (D_n)) &= 0 \\ V^k(x, (D_n)) &= I\{x^t(N) = 0\} + \beta \min_\delta \textstyle\sum_{y \in \mathcal{X}} p_{xy}(\delta, (D_n)) V^{k-1}(y) \forall k \geq 1 \end{aligned} \tag{4.6}$$

We are now ready to formally state the proposition.

Proposition 16. *Given a buffer with length N and a discount rate $0 \leq \beta < 1$, an optimal downloading policy $\hat{\delta}$ will always choose to download the chunk in buffer position $N-1$ if such a download is possible. Among the class of permutation-based policies, this implies if the optimal policy permutation $\hat{\pi}$, then $\hat{\pi}_1 = N - 1$.*

Proof. We use a stochastic coupling argument to prove the claim. We denote buffer position i to be B_i. Let δ be some downloading policy such that there exists some (D_n) so that δ will not download $B_N - 1$ when such a download is possible at time slot t_0. We show that it is always possible to construct a policy $\tilde{\delta}$ that downloads B_{N-1} whenever possible, and that $C^k\left(x^t, \delta, (D_n)\right)$ is stochastically dominated by $C^k\left(x^t, \tilde{\delta}, (D_n)\right)$.

Assume there are two users, a and b, who have identical buffer states x^t, and are exposed to the same sequence (D_n). User a adopts downloading policy δ, while user b adopts $\tilde{\delta}$.

Construct $\tilde{\delta}$ as follows:

1. Starting at t_0, $\tilde{\delta}$ downloads exactly the same chunk as δ in each iteration, until the first round, say $t = k$, when the set of possible downloading locations D_k is $\{B_j, B_{N-1}\}, j < N-1$, where B_j is assigned a higher priority than B_{N-1} according to δ.

2. At time $t = k$, $\tilde{\delta}$ will choose to download chunk c that is now at position B_{N-1}, while δ will download chunk d that is now at position B_j.

3. From time $t = k+1$ to $t = k + (N-1-j)$, $\tilde{\delta}$ continue to mimic the downloading choices of δ, except when the only possible downloading for user b is $B_{j+(t-k)}$, and no download is possible for user a, since a and b should have the same buffer state except for location $B_{j+(t-k)}$. In this case user b ($\tilde{\delta}$) will download chunk d, while user a does nothing.

4. After time $t = k + (N - 1 - j)$, by which time chunk d would have been played, $\tilde{\delta}$ becomes identical to δ for the rest of the time.

Because time t goes to infinity, the event described in Step 1 is going to happen with probability 1. Note that the randomness of V is solely contributed by the randomness of the sequence (D_n). Denote the probability space on which (D_n) is defined as $\Omega_0 = \{B_2, B_3, ..., B_{N-1}\}^\infty$. Create a partition $\{\Omega_1, \Omega 2\}$ of Ω_0, such that Ω_1 contains all sequences (D_n), based on which the chunk d will be downloaded by user b as described in Step 3.

Observe that:

1.
$$C\left(x^t, \delta, (D_n)\right) - C\left(x^t, \tilde{\delta}, (D_n)\right) = \beta^{k-t} > 0, \forall (D_n) \in \Omega_1 \tag{4.7}$$

   This is true since chunk d was later downloaded by b. Overall user a was charged an additional cost of β^{k-t} compared to user b for missing chunk c.

2.
$$C(x^t, \delta, (D_n)) - C\left(x^t, \tilde{\delta}, (D_n)\right) = \beta^{k-t} - \beta^{k-t+N-1-j} \geq 0, \forall (D_n) \in \Omega_2 \tag{4.8}$$

   In this case user a missed chunk c while b missed chunk d. However, the cost for missing d was incurred at a later time, which led to a lower or equal discounted cost compared to missing chunk c.

We have shown that $C(x^t, \delta, (D_n))$ presents a higher cost than $C\left(x^t, \tilde{\delta}, (D_n)\right)$ on any sample path of future downloadable sets (D_n). Therefore, an optimal downloading policy must download the piece at buffer position B_{N-1} whenever possible. □

Note:The above proof is derived with the assumption that the discount factor $\beta < 1$, and hence the infinite-horizon cost-to-go converges. If we were to make a similar argument about the optimal downloading policies with $\beta = 1$, such that the focus is only on the average downloading rate, the same argument applies, but only when the time horizon k is finite since the limit $lim_{k\to\infty} C(x, \delta, (D_n))$ may not exist if $\beta = 1$.

4.3 Is EDF Always Optimal?

While EDF is clearly not an optimal strategy if all users were to follow it (shown in simulations from the previous chapter), since it suppresses the spreading of rare chunks within the network, could it be optimal if a selfish user is to use it against any $fixed$ target distribution? Proposition 16 in some sense states the absolute importance of the buffer positions $N - 1$ in downloading, and intuitively speaking, a peer would prefer to download chunks that are closer to the playback deadline (buffer location N), since there is less time left to download such pieces. This leads to the following conjecture:

Conjecture 17. *(**Optimality of EDF**) It is an optimal downloading policy to always download the chunk with the largest buffer index possible given any fixed target distribution.*

Unfortunately, it turns out such a conjecture is not true. The following proposition disproves the conjecture and offers some insight on when a peer would prefer not to download the chunk with the earliest deadline.

Proposition 18. *Let the length of buffer $N = 5$. Let the target's buffer-state distribution for locations $\{1, 2, 3, 4\}$ consist of the product of four independent Bernoulli random variables with coefficients $p_1 = p_2 = \delta, p_3 = p_4 = \gamma$. Then EDF is NOT an optimal strategy if both of the following two conditions hold:*

1. $1 > \gamma >> \delta \geq 0$
2. $\gamma > \sqrt{\frac{1}{\beta} - 1}$, *where β is the discount factor.*

Proof. Suppose in the present time slot the user's buffer state is $x = 0011/1$, where the forward slash ($/$) is used to simply remind us that no download is permitted in the last (N-th) buffer location. Suppose also that both chunks C_1 and C_2 (in locations 1 and 2, respectively) are available at the target in this time slot. We argue that the optimal strategy will be to download C_1 rather than C_2, which disagrees with EDF.

Note that with a discount factor $\beta < 1$, $\lim_{k\to\infty} V^k(x) = V(x), \forall x \in \mathcal{X}$. We would like to show that the state resulted by downloading chunk C_1 has a lower cost-to-go compared to downloading chunk C_2, or equivalantly:

$$V(0101/1) < V(0011/1). \tag{4.9}$$

We also have:

$$\begin{aligned} 0011/1 &= SD_2(0011/1) \\ 0101/1 &= SD_1(0011/1) \end{aligned} \tag{4.10}$$

where S is the shift operator and $D_{(\cdot)}$ is the download operator. Note that since the downloading process follows the charge-download-shift cycle, the most significant (or left-most) bit of the state vector x should always be zero in between two adjacent time slots.

Since we are interested in disproving the conjecture that EDF's optimality, let us assume that EDF is indeed optimal for all other states other than the case where user has state $x = 0011/1$. We also assume that since δ is very small, the probability of getting anything from position 1 or 2 in one time slot is negligible. Expanding the expression of V, based on the assumption that EDF is optimal for all other states we obtain:

$$\begin{aligned} V(0101/1) &\overset{\frac{\delta}{\gamma}\to 0}{\approx} I\{x_5 = 0\} + \gamma\beta V(0011/1) + (1-\gamma)\beta V(0010/1) \\ &\overset{\frac{\delta}{\gamma}\to 0}{\approx} \beta^2 \left\{ \left[1 - (1-\gamma)^2\right] V(0001/1) + (1-\gamma)^2 V(0001/0) \right\} \\ &= \beta^2 \left\{ \left[1 - (1-\gamma)^2\right] V(0001/1) + (1-\gamma)^2 \left(V(0001/1) + 1\right) \right\} \\ &= \beta^2 \left[V(0001/1) + (1-\gamma)^2 \right]. \end{aligned} \tag{4.11}$$

We also have for state 0011/1":

$$
\begin{aligned}
V(0011/1) &\overset{\frac{\delta}{\gamma}\to 0}{\approx} I\{x_5 = 0\} + \beta \cdot V(0001/1) \\
&\overset{\frac{\delta}{\gamma}\to 0}{\approx} \beta^2 \left[\gamma V(0001/1) + (1-\gamma) V(0000/1)\right] \\
&= \beta^2 \left[V(0001/1) + (1-\gamma)\left(V(0000/1) - V(0001/1)\right)\right]. \qquad (4.12)
\end{aligned}
$$

Further expanding the terms, we have:

$$
\begin{aligned}
V(0001/1) &\overset{\frac{\delta}{\gamma}\to 0}{\approx} 0 + \gamma\beta V(0001/1) + (1-\gamma)\beta V(0000/1) \qquad (4.13) \\
V(0000/1) &\overset{\frac{\delta}{\gamma}\to 0}{\approx} 0 + \gamma\beta V(0000/1) + (1-\gamma)\gamma\beta V(0001/0) + (1-\gamma)^2 \beta V(0000/0)
\end{aligned}
$$

Subtracting $V(0000/1)$ by $V(0001/1)$, we get:

$$
\begin{aligned}
&\frac{1}{\beta}\left(V(0000/1) - V(0001/1)\right) \\
= \;& (2\gamma - 1)V(0000/1) + \left[(1-\gamma)\gamma V(0001/0) - \gamma V(0001/1) + (1-\gamma)^2 V(0000/0)\right] \\
= \;& (2\gamma - 1)V(0000/1) + (1-\gamma)\gamma\left[V(0001/1) + 1\right] - \gamma V(0001/1) + (1-\gamma)^2 V(0000/0) \\
= \;& (2\gamma - 1)V(0000/1) - \gamma^2 V(0001/1) + (1-\gamma)\gamma + (1-\gamma)^2 V(0000/0) \\
= \;& (2\gamma - 1)V(0000/1) - \gamma^2\left[V(0000/1) - \left(V(0000/1) - V(0001/1)\right)\right] \\
& + (1-\gamma)\gamma + (1-\gamma)^2 V(0000/0) \qquad (4.14)
\end{aligned}
$$

Rearranging (4.14) and moving the term $(V(0000/1) - V(0001/1))$ to the left-hand side, we get:

$$
\begin{aligned}
&\left(\frac{1}{\beta} - \gamma^2\right)\left(V(0000/1) - V(0001/1)\right) \\
= \;& -(1-\gamma)^2 V(0000/1) + (1-\gamma)^2 V(0000/0) + (1-\gamma)\gamma \\
= \;& -(1-\gamma)^2 V(0000/1) + (1-\gamma)^2\left(V(0000/1) + 1\right) + (1-\gamma)\gamma \\
= \;& 1 - \gamma \qquad (4.15)
\end{aligned}
$$

from which we obtain

$$
V(0000/1) - V(0001/1) = \frac{\beta(1-\gamma)}{1-\beta\gamma^2}. \qquad (4.16)
$$

Combining (4.11) and (4.12) and by using (4.16), we arrive at the final inequality:

$$
\begin{aligned}
V(0011/1) - V(0101/1) &\overset{\frac{\delta}{\gamma}\to 0}{\approx} \beta^2 \left[(1-\gamma) \left(V(0001/1) - V(0000/0) \right) - (1-\gamma)^2 \right] \\
&= \beta^2 \left[(1-\gamma) \frac{\beta(1-\gamma)}{1-\beta\gamma^2} - (1-\gamma)^2 \right] \\
&= \beta^2 \left[\frac{(1-\gamma)^2}{1-\beta\gamma^2} (\beta - 1 + \beta\gamma^2) \right] \\
&> \beta^2 \left[\frac{(1-\gamma)^2}{1-\beta\gamma^2} \left(\beta - 1 + \beta \left(\frac{1}{\beta} - 1 \right) \right) \right] \\
&= 0 \qquad (4.17)
\end{aligned}
$$

Since the buffer state $x = 0011/1$ has a strictly higher cost-to-go compared to state $x = 0101/1$, the EDF policy is hence not optimal.

□

Chaper 5

Conclusion

In this thesis we examined a set of problems concerning the performance of a peer-to-peer live-streaming network. The majority of our results were obtained using the marginal probability model, while much of the ongoing research is oriented on the user state model with a problem formulation using dynamic programming.

Through our presentation, the reader may have already come to notice the trade-off between the simplicity and accuracy every model entails. For example, despite the fact that the marginal probability model operates on an independence assumption that is not true in practice, the error is generally small is in some situations tolerated given the resulted simplicity. On the other hand, the model is no longer capable of providing us with further insight as we begin to investigate the optimal downloading policies for a selfish user in a more rigorous setting, in which case the user state model and the related concept of dynamic programming, although a bit more complex, came in as a very powerful tool.

A future direction of research may be in the area of Nash Equilibrium downloading policies. It would be interesting to know that assuming *every* user is selfish, such that they all constantly adjust their downloading policies to minimize cost-to-go against the chunk distribution of the surrounding peers, what would be the convergent downloading policy δ^* and chunk distribution $\mathcal{F}^*$ be so that a peer will be penalized with a higher cost-to-go if it chooses to deviate from δ^*. This analysis would give us some sense of a worst-case performance of the network, since the enforcement of a peer's choice of chunks is generally more difficult to accomplish compared to imposing incentive strategies such as the tit-for-tat requirement described in Section 3.3.

Chaper 6

References

[1] Yipeng Zhou, Dah-Ming Chiu, and John C. S. Lui, "A Simple Model for Analyzing P2P Streaming Protocols," *ICNP 2007*, pp. 226–235.

[2] Dah-Ming Chiu et al., "Optimizing Piece Selection in data-driven P2P Streaming Systems," Preprint.

[3] Bruce Hajek, "An Exploration of Random Processes for Engineers," *ECE 534 Course Notes*, January 2009.

[4] Bruce Hajek, "Communication Network Analysis," *ECE 567 Course Notes*, December 2006.