On the Resilience-Complexity Tradeoff of Network Coding in Dynamic P2P Networks

Di Niu, Baochun Li
Department of Electrical and Computer Engineering
University of Toronto
{dniu, bli}@eecg.toronto.edu

***Abstract*—Most current-generation P2P content distribution protocols use fine-granularity *blocks* to distribute content to all the peers in a decentralized fashion. Such protocols often suffer from a significant degree of imbalance in block distributions, such that certain blocks become rare or even unavailable, adversely affecting content availability. It has been pointed out that randomized network coding may improve block availability in P2P networks, as coded blocks are equally innovative and useful to peers. However, the computational complexity of network coding mandates that, in reality, network coding needs to be performed within *segments*, each containing a subset of blocks. In this paper, using both theoretical analysis and simulations, we quantitatively evaluate how segment-based network coding may improve resilience to peer dynamics and content availability. The objective of this paper is to explore the fundamental tradeoff between the resilience gain of network coding and its inherent coding complexity. We introduce a differential equations approach to quantify the resilience gain of network coding as a function of the number of blocks in a segment, as well as various other tunable parameters. We conclude that a small number of blocks in each segment is sufficient to realize the major benefits of network coding, with acceptable coding complexity.**

I. Introduction

Peer-to-peer (P2P) content distribution has become the *de facto* standard in current-generation content distribution protocols. The basic idea in P2P content distribution protocols is to segment large volumes of data (usually hundreds of megabytes or even gigabytes) into fine-granularity *blocks*, and then distribute these blocks in an efficient manner by letting peers exchange them with one another.

In reality, however, P2P protocols often suffer from severe peer dynamics (*i.e.,* arrivals and departures), as peers are inherently unreliable. As the fundamental philosophy of peer-to-peer protocols is to use resources on the peers to store blocks of content, content distribution sessions may continue to proceed even when the original peers (sometimes referred to as *seeds*) are no longer available. However, in these cases, blocks may become rare or even unavailable when peers arrive and depart frequently with short lifetimes. Such significant degrees of imbalance with respect to block availability — henceforth referred to as *block variation* — will adversely affect the availability of content, leading to longer downloading times at each peer.

As end hosts at the edge of the Internet possess abundant computational resources with modern processors, it is natural to take advantage of the power of *network coding* in P2P applications, by allowing end hosts to not only forward and replicate, but to code as well. *Network coding* has been originally proposed in information theory [1], [2], [3], and has been more recently used to improve the resilience to peer dynamics in P2P content distribution protocols [4], leading to shorter downloading times. Intuitively, one may observe that, since network coding distributes coded blocks rather than original blocks, and all coded blocks are equally innovative and useful to any peer, the challenge of locating rare original blocks may indeed be addressed.

However, network coding may not realize its benefits without introducing significant computational complexities at peers. It has been shown in recent work [5] that, network coding may not be computationally feasible if one is to code more than a few hundred blocks, even with modern processors! In order to reduce coding complexity, Chou *et al.* [6] has proposed the concept of *group network coding*, which performs coding on the blocks within the same *group* or *segment*, while each segment contains a prescribed (and arguably small) number of blocks. Though group network coding helps to reduce coding complexity, its negative effects on the resilience to peer dynamics — the main advantage of using network coding in the first place in P2P networks — are not fully understood.

Let us consider two extremes of P2P protocol design. The first one does not use network coding at all, and the second uses network coding across all existing blocks. If we consider the number of blocks in each segment (referred to as the *segment size*) in group network coding, we may observe that the first extreme corresponds to a segment size of one, with as many segments as blocks, while the other extreme corresponds to the case of grouping all blocks into the same segment. Intuitively, the degree of resilience to peer dynamics that network coding has to offer improves as we increase the segment size; but the coding complexity increases as well. If we vary the segment size when network coding is used, we are fundamentally moving from one extreme to another, and making our choice in the challenge of *resilience-complexity* tradeoff of network coding in P2P networks. It would be best if we may operate with an appropriate segment size to enjoy most of the resilience advantage of using network coding, but with acceptable coding complexity.

Motivated by our curiosity on choosing the "sweet spot" in the resilience-complexity tradeoff, we quantitatively evaluate the content availability with different segment sizes, using both theoretical analysis and simulations. In our theoretical analysis, we consider large-scale dynamic P2P systems, where each peer is to have a random lifetime following an arbitrary yet general distribution. We operate a linear system of differential equations, and derived closed-form results with respect to the

The completion of the research was made possible thanks to Bell Canada's support through its Bell University Laboratories R&D program.

variations of block availability. We use simulations to not only substantiate our theoretical conclusions, but also shed further insights into the problem in a wide range of scenarios. To the best of our knowledge, such a resilience-complexity tradeoff introduced by network coding in P2P content distribution has never been studied with theoretical rigor in previous work.

The remainder of this paper is organized as follows. In Sec. II, we present related work. In Sec. III, we present our system model, formulate the problem and outline the main theoretical results. In Sec. IV, we derive the differential equations on which we obtain results regarding content availability and block variations. In Sec. V, we present theoretical results regarding steady state block variation when data sources are available and content availability in the absence of data sources. In Sec. VI, we carry out simulations to corroborate our theoretical results. We conclude the paper in Sec. VII.

II. Related Work

The landmark papers on randomized network coding by Ho *et al.* [7] and Chou *et al.* [6] have claimed that randomized network coding can be designed to be robust to random packet loss, delay, as well as any changes in network topology and capacity. Avalanche [4], [8] has further proposed that randomized network coding can be used for elastic content distribution. However, it has also been shown [5] that the coding complexity escalates when an increasing number of blocks are used in network coding, and that even with modern processors, coding more than a few hundred blocks may not be computationally feasible.

In Wu [9], it has been argued that a key advantage of network coding is its inherent ability to adapt to network dynamics, both to ergodic changes such as random packet loss and to non-ergodic changes such as link failures. Koetter *et al.* [3] also discussed robust networking and analyzed the resilience of network coding to non-ergodic link failures. Lun *et al.* [10] theoretically analyzed the benefit of network coding on a directed acyclic hypergraph with lossy links. Acedanski *et al.* [11] showed through analysis that with network coding, a peer downloading a file may randomly connect to fewer other peers to retrieve the entire file. The relationship between block selection policies and the imbalance among different blocks has been discussed for P2P content distribution without network coding in Fan *et al.* [12]. In this paper, we focus on the resilience-complexity tradeoff of network coding, when the number of blocks in a segment varies from one extreme to another. To our knowledge, this is a first attempt to quantitatively evaluate the resilience of network coding to block variations due to peer dynamics.

III. Background, Model and Main Results

Modern P2P bulk content distribution systems (e.g., BitTorrent [13]) are organized as an application-layer overlay of peers, each with a number of neighbors. A peer is referred to as a *server* if it owns a complete copy of the content of interest, otherwise it remains as a *user*. Data exchanges may only occur between a peer and its neighbors. Normally, a peer only uploads to a small number (e.g., less than 5) of its neighbors at a time which are called its downstream peers, since excessive concurrent uploading may negatively affect throughput [14]. However, a peer may still frequently change its downstream peers, either because it intentionally does so to take advantage of idle capacities, or due to peer dynamics.

We now present a framework that allows us to analyze P2P content distribution with variable coding complexities. A large file of size F bytes (usually on the order of hundreds of Megabytes or several Gigabytes) is to be broadcast to every online peer. The content is segmented into G segments, each of which are further broken into m blocks, referred to as the *segment size*. Thus, there are $M = G \cdot m$ different original blocks in the content, each of size $k = F/M$ bytes. In group network coding [6], random linear coding (RLC) is applied to each segment of m blocks. Assume segment i has original blocks $\mathbf{B^{(i)}} = [B_1^i, B_2^i, \ldots, B_m^i]$, then a coded block b from segment i is a linear combination of $[B_1^i, B_2^i, \ldots, B_m^i]$ in the Galois field GF(2^8). Coding operation is not limited to the source: if a peer (including the source) has l $(l \leq m)$ coded blocks of segment i $[b_1^i, b_2^i, \ldots, b_l^i]$, when serving another peer p, it independently and randomly chooses a set of coding coefficients $[c_1^p, c_2^p, \ldots, c_l^p]$ in the Galois field GF(2^8), and then encodes all its blocks from segment i, and produces one coded block x of k bytes: $x = \sum_{j=1}^{l} c_j^p \cdot b_j^i$.

A coded block x is *self-contained*, in that the coding coefficients used to encode *original blocks* to x are embedded in the header of the coded block. As soon as a peer has received a total of m coded blocks from segment i $\mathbf{x} = [x_1^i, x_2^i, \ldots, x_m^i]$, which are linearly independent, it will be able to decode segment i with coding coefficients embedded in each of the m coded blocks. To decode segment i, we first need to compute the inverse of the $m \times m$ coefficient matrix $\mathbf{A_i}$ using Gaussian elimination, which requires $O(m^3)$ operations (or $O(m^2)$ operations per input block). To obtain the original m blocks $\mathbf{B^{(i)}}$, it then needs to multiply $\mathbf{A_i}^{-1}$ and $\mathbf{x}$, which takes $m^2 \cdot k$ multiplications of two bytes in GF(2^8), which runs in time $O(m^2)$ (or $O(m)$ operations per input block). It turns out the latter cost dominates the overall decoding time, although the first phase has a higher computational complexity in terms of m. This is because the cost of the latter phase also depends on the block size k, which is usually on the order of Kilobytes. Naturally, the overall decoding complexity increases as the segment size m increases.

Assume there are N user peers and N_s servers online, each with an average upload capacity μ and a separate downlink of sufficiently large capacity, then a peer will upload at a rate of $\lambda = \mu/k$ blocks on average. With respect to reducing block variation in P2P networks, network coding and block balancing schemes (e.g., the local rarest first policy in BitTorrent [13]) interact in complicated ways. It is therefore impossible to assess the resilience of network coding without reference to block balancing schemes. However, in order to focus on the effect of network coding, we will use a gossiping protocol as the reference. Specifically, whenever a peer is to upload a block, it randomly chooses a downstream peer from all other peers. (This assumption can be relaxed to randomly choosing a peer from its neighborhood, as long as the neighbors are uniform samples of the entire network.) After that, block uploading is performed in three steps: segment reconciliation, encoding, and transmission. First, the peer will compare its buffer with that of the downstream peer p to find out which segments that it has are needed by p. As a result, a difference set of segments is formed. It then randomly chooses a segment in the difference set and encodes all the blocks from that segment. Finally, it transmits the encoded block to p.

To model peer dynamics, we fix the online peer number while allowing peers to join and leave the network frequently. Specifically, we assume that there are always N users and N_s servers simultaneously online, even though user departures and joins occur constantly. Servers are always online. Each user has a random lifetime $\boldsymbol{L}$. A user will leave the network when its lifetime has expired. In the meantime, a new user with a random lifetime $\boldsymbol{L}$ as well will join the network to replace the departed user. No peer will return after it leaves. A similar model for churn can be found in [15]. As will be shown, by modeling peer dynamics this way, the total number of blocks in the network will remain constant while the distribution of blocks from each segment could be changing in time. This enables us to focus on studying the block variation. In fact, the assumption on a fixed number of online peers can be relaxed to accommodate slow changes of N and N_s in time. As long as such changes in $N(t)$ and $N_s(t)$ take place at a much slower rate than the average upload rate of a peer, the analysis will hold the same except that N and N_s are considered as functions of t in the results. At a given observation time t_o, we also define a peer's *age* $\boldsymbol{A}$ as the time till t_o since it joined. In our analysis, the peer age will be an important parameter (note its difference from the lifetime $\boldsymbol{L}$).

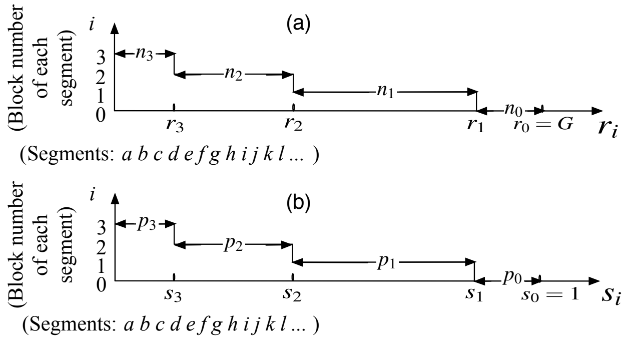
(Block number of each segment)
i
(a)
3
2
1
0
n3
n2
n1
n0
r3
r2
r1
r0 = G
ri
(Segments: a b c d e f g h i j k l ...)
(b)
p3
p2
p1
p0
s3
s2
s1
s0 = 1
si
(Segments: a b c d e f g h i j k l ...)

Fig. 1. An illustration for the notations $p_i(t)$, $s_i(t)$, $n_i(t)$ and $r_i(t)$.

Our main goal is to derive the steady-state distribution of the number of blocks in each segment when N and M approaches infinity. If we denote by a random variable $\boldsymbol{I}$ the number of blocks each segment has in users with mean μ_I and variance σ_I^2, we are interested in deriving the distribution for $\boldsymbol{I}$. The derivation of this block distribution will allow us to evaluate important metrics such as block variation, download time, content loss rate and content lifetime. To characterize the distribution of $\boldsymbol{I}$, we therefore introduce a system of important notations that represent the system's state. Since servers always have complete copies of the content, we are only interested in knowing the block statistics in users:

- ▷ $n_i(t)$: the number of segments which have i blocks in all the users. $\sum_{i=0}^{\infty} n_i(t) = G$ for any t.
- ▷ $p_i(t)$: the fraction of segments which have i blocks in all the users. $p_i(t) = n_i(t)/G$.
- ▷ $r_i(t)$: the number of segments which have at least i blocks in all the users. $n_i(t) = r_i(t) - r_{i+1}(t)$.
- ▷ $s_i(t)$: the fraction of segments which have at least i blocks in all the users. We have $s_i(t) = r_i(t)/G$ and $p_i(t) = s_i(t) - s_{i+1}(t)$.

These notations, illustrated in Fig. 1, will be the main variables of interest in our analysis with differential equations. Another helpful way to view Fig. 1 is that the vertical axis i represents the number of blocks each segment has in all the users, and the horizontal axis represents segments arranged in the decreasing order of i. It is not hard to see that as the segment number $G \to \infty$, $p_i(t) = n_i(t)/G$ will approach the PMF of the variable $\boldsymbol{I}$. These notations cover both non-coding and coding cases as the segment size m varies. For simplicity, we may omit t in the following context. From Fig. 1, we also get the following simple facts: *1) the total number of blocks in users at time t is $Y(t) = \sum_{i=1}^{\infty} r_i(t)$, and 2) the average number of blocks each segment has in users at time t is $\sum_{i=1}^{\infty} s_i(t) = Y(t)/G$*, which equals to the area under the line in Fig. 1(b).

The state of the system could be represented by the vector $\boldsymbol{S}(t) = (s_0(t), s_1(t), \ldots)$ or $\boldsymbol{P}(t) = (p_0(t), p_1(t), \ldots)$. If we use $\boldsymbol{S}(t)$ to denote the system's states for instance, at any given time t, the future state of the system is solely decided by $\boldsymbol{S}(t)$, in that both block increase and block loss of each segment depend merely on the current values of $s_0(t), s_1(t), \ldots$, given the memoryless property of the gossiping protocol and of the exponential block upload time. Therefore, vector $\boldsymbol{S}(t)$ represents a density dependent family of jump Markov processes [16], with the number of segments G being the total population size and $(s_0, s_1, \ldots)$ being densities. Apparently, once we obtain the expressions for $\boldsymbol{S}(t)$, we can obtain all the block statistics at any time t. However, since such Markov processes are too complicated to handle technically, we will introduce in the next section a system of differential equations which asymptotically approximate the corresponding Markov processes described above with an arbitrarily small error.

To evaluate degrees of imbalance with respect to block availability of different segments, we define an important parameter called *block variation* as:

$$\gamma_I^2 = \frac{\text{var}(\boldsymbol{I})}{\text{E}^2\{\boldsymbol{I}\}} = \frac{\sum_{i=0}^{\infty} i^2 p_i - (\sum_{i=0}^{\infty} i \cdot p_i)^2}{(\sum_{i=0}^{\infty} i \cdot p_i)^2}, \qquad (1)$$

which is essentially the variance of the number of blocks each segment has in users divided by the square of its mean. var($\boldsymbol{I}$) alone does not serve as a good indicator of block variation, in that its effectiveness in assessing block imbalance may be biased by the value of E$\{\boldsymbol{I}\}$. Thus, we adopt the above definition of block variation, which is inspired by the typical fairness measure in resource allocation [17]. Block variation γ_I^2 has the nice property that its value always lies between 0 and ∞, and that it is 0 for the balanced block distribution when each segment has the same number of blocks in the network. As we will show through simulations, download time is directly related to block variation, which is a good indicator of the availability of different segments. As a major theoretical contribution of this paper, we have derived a surprisingly concise yet powerful formula which sheds many insights into network coding's behaviour in face of churn:

Theorem: For large N, M, in steady state network, the block variation γ_I^2 is inversely proportional to m, the number of blocks within a segment for coding:

$$\gamma_I^2 = \frac{F}{N_s m \mu \overline{A}}, \tag{2}$$

where $\overline{A}$ is the average age of a user in a steady state network that will be determined by Lemma 1.

As the segment size increases, the block variation is subdued. An intuition behind the benefit of network coding is that without network coding ($m = 1$), departing peers may take away important blocks that have already become rare in the network, while with pure network coding ($m = M$), all the blocks are equally useful and thus there is no variation of block availability at all. As segment size m grows, the total number of distinct segments is reduced. Hence, intuitively, one may observe that different segments are distributed in a more balanced way in face of churn.

IV. Characterizing System States with Differential Equations

In this section, we formulate a set of differential equations to characterize the network's states by asymptotically approximating the underlying Markov processes. The correctness of this approach was proved by Kurtz [18]. As an application, it was then successfully applied into the solution to a Markov queuing model in load balancing problems by Mitzenmacher [16], [19]. With proper simplifications at several points, we are able to use this method to model the seemingly complex system of P2P content distribution with node churn. The main results regarding content availability drawn from the differential equations are presented in the next section.

First, we will show the proposed network model has an equilibrium, in which the total number of blocks in the network stays constant. But before showing this in Lemma 2, let us first cite a useful lemma proposed by [15] and demonstrated again in [20].

Lemma 1. Let $\boldsymbol{L}$ denote a peer's lifetime with mean $\overline{L}$ and variance σ^2. Given an observation time t_o, let random variable $\boldsymbol{A}$ denote a peer's *age* at t_o, with expectation $E[\boldsymbol{A}] = \overline{A}$. If N is fixed, then as $t_o \to \infty$, the probability density function of $\boldsymbol{A}$ is given by $f_A(x) = [1 - F_L(x)]/\overline{L}$. Moreover, $E[\boldsymbol{A}] = \overline{A} = (\overline{L}^2 + \sigma^2)/2\overline{L}$.

It is easy to check that if the peer lifetime $\boldsymbol{L}$ follows an exponential distribution, the distribution of peer age $\boldsymbol{A}$ will be exactly the same as that of $\boldsymbol{L}$. In the following lemma, we show that after the network has evolved for a sufficiently long time, the total number of blocks in users will almost always remain constant.

Lemma 2. Let $\boldsymbol{Y(t)}$ denote the total number of blocks in users from all the segments at a given time t for t sufficiently large, then $\frac{\boldsymbol{Y(t)}}{(N+N_s)} \to \lambda\overline{A}$ in probability, i.e., for any $\epsilon > 0$,

$$P\{|\frac{\boldsymbol{Y(t)}}{(N+N_s)} - \lambda\overline{A}| > \epsilon\} \to 0, M \to \infty, N = \alpha M, N_s = \alpha_s M. \tag{3}$$

Proof. Please refer to our technical report [21]. □

From Lemma 2, we know that, for a large segment number G, the average number of blocks each segment has in users is $\sum_{i=1}^{\infty} s_i(t) = \boldsymbol{Y}(t)/G = (N+N_s)\lambda\overline{A}/G = (\alpha+\alpha_s)m\lambda\overline{A}$.

We now give the system of ODEs that characterize the block distributions of different segments. **Under the condition that $M \to \infty$, $N \to \infty$, $N_s \to \infty$, $N/M \to \alpha$, $N_s/M \to \alpha_s$, and m remains finite,** though α_s is often far less than α and can also be 0, **the block distributions are characterized by the following system of ODEs:**

$$\begin{aligned} \overline{A} \cdot \frac{\mathrm{d}s_i}{\mathrm{d}t} &= \alpha_s m \lambda \overline{A} \cdot p_{i-1} + \frac{\alpha}{\alpha+\alpha_s}(i-1)p_{i-1} - i \cdot p_i \quad \forall i \geq 1 \\ s_0 &= 1 \end{aligned} \tag{4}$$

Let us explain the reasoning behind these ODEs. We consider a small time interval Δt and decide the expected change of r_i in Δt, denoted by Δr_i. We denote by $I(r_i)$ the expected increase in r_i due to the upload from users, by $I_s(r_i)$ the expected increase in r_i due to the upload from servers, and by $D(r_i)$ the expected decrease in r_i due to block losses caused by peer departures. We have $\Delta r_i = I_s(r_i) + I(r_i) - D(r_i)$.

We first determine $I_s(r_i)$. Since the time to upload a block is exponentially distributed, the expected number of blocks the servers have uploaded is $N_s\lambda\Delta t$. Now we wish to know how many of these $N_s\lambda\Delta t$ blocks contribute to the increase in r_i. From Fig. 1, we see that r_i increases by one if and only if a segment with $i-1$ blocks increases a block. We choose $N_s, M, \Delta t$ such that when $N_s \to \infty, M \to \infty, N_s/M \to \alpha_s$ and $\Delta t \to 0$, the total number of blocks servers have uploaded $N_s\lambda\Delta t$ will also approach to infinity, but is less than the segment number $G = \frac{M}{m}$ (e.g., letting $\Delta t = \Theta(\frac{1}{\sqrt{N_s}})$ can achieve this). With this requirement, each segment can either increase 1 block or not increase at all in Δt. Since all the segments are perfectly balanced in servers, each segment has an equal chance of being chosen, as all the blocks are largely needed by most peers, which are highly dynamic. Thus, we obtain

$$I_s(r_i) = N_s\lambda\Delta t \cdot p_{i-1} \tag{5}$$

Note that to simplify the analysis, we have implicitly assumed that no linear dependency will occur when a peer is updating another with network coding. This assumption is reasonable because according to Lemma 2.1 in [22], a random linear combination of all the blocks from the same segment at a peer p is useful to another randomly chosen peer in the network with probability at least $1 - 1/q$ if network coding is done in $\mathbb{F}_q$. And this argument is true regardless of whether peer p is a server or a user. In this paper, the field size q has been assumed to be 2^8.

We then consider the value of $I(r_i)$. Similarly, the total block increase of all the segments due to users' upload is $N\lambda\Delta t$. However, the increase in r_i is no longer $N\lambda\Delta t \cdot p_{i-1}$, in that each segment does not enjoy an equal chance of being chosen by the users. On the contrary, however, the more blocks a segment has in all the users, the more frequently it will be encoded and uploaded. Therefore,

$$I(r_i) = N\lambda\Delta t \cdot \frac{(i-1)n_{i-1}}{\boldsymbol{Y}} \rightarrow \frac{\alpha}{\alpha+\alpha_s}\frac{\Delta t(i-1)n_{i-1}}{\overline{A}}, \quad (6)$$

where $\boldsymbol{Y}$ is the total number of blocks in users given by Lemma 2. In fact, the experimental results in Sec. VI have substantiated this observation.

Finally, we determine $D(r_i)$. First, we know that the total loss of blocks in Δt is $(N_s+N)\lambda\Delta t$, because the total number of blocks remains unchanged by Lemma 2. Similarly, r_i decreases by one if and only if a segment with i blocks loses a block. And the more blocks a segment has in all the users, the more blocks it loses due to peer departures. Hence, we have

$$D(r_i) = (N_s+N)\lambda\Delta t \cdot \frac{i \cdot n_i}{\boldsymbol{Y}} \rightarrow \frac{\Delta t i \cdot n_i}{\overline{A}} \quad (7)$$

Substituting $I_s(r_i), I(r_i), D(r_i)$ into $\Delta r_i = I_s(r_i) + I(r_i) - D(r_i)$, we get

$$\Delta r_i = N_s\lambda\Delta t \cdot p_{i-1} + \frac{\alpha}{\alpha+\alpha_s}\frac{\Delta t(i-1)n_{i-1}}{\overline{A}} - \frac{i \cdot n_i}{\overline{A}} \cdot \Delta t \quad (8)$$

Dividing by $G \cdot \Delta t$ on both sides, we get the system of linear differential equations (4).

We have omitted the rigorous proof of the concentration of (4) around their underlying Markov processes, since such a proof deviates from the thesis of this paper and can be done in a standardized way [18], [16]. According to Mitzenmacher [16], [19], one necessary condition for these differential equations to hold in theory is that the average number of blocks each segment has in users $\sum_{i=1}^{\infty} s_i(t)$ remains finite at all times, which implies cases with severe churn. This condition is satisfied in our model, since $\sum_{i=1}^{\infty} s_i(t) \rightarrow (\alpha+\alpha_s)m\lambda\overline{A}$ in probability, which we require to remain finite.

V. Quantifying the Resilience of Network Coding

In this section, we operate on the system of ODEs (4) to analyze the resilience of network coding in both the steady state when servers are online, and the transient stage after servers' departure.

A. Steady State Block Variation and Block Distribution

Assuming the presence of servers, let us now determine the block variation and block distribution in steady state networks. Our main findings are that the steady-state block variation is inversely proportional to the segment size for network coding, and that the steady-state block distribution follows a form of *negative binomial distribution*, which can be well approximated by a Gaussian distribution. The block variation is a good indicator of the download time, which we will show in the simulations. And the block distribution in steady state not only offers an intuitive concept with respect to the imbalance of block availability, but also serves as a starting point for analyzing content availability in the absence of servers.

Let us denote the steady-state solutions to (4) by $\overline{p}_0, \overline{p}_1, \ldots, \overline{p}_G$. By setting $\frac{\mathrm{d}s_i}{\mathrm{d}t} = 0$ in the differential equations (4), we get $\overline{p}_i = [B + \beta(i-1)] \cdot p_{i-1}/i$, where $B = \alpha_s m\lambda\overline{A}$, $\beta = \frac{\alpha}{\alpha+\alpha_s}$. Hence, the steady-state block distribution is given by,

$$\overline{p}_i = \overline{p}_0 \cdot \prod_{j=1}^{i}(\beta + \frac{B-\beta}{j}), \quad \text{for } i \geq 1. \quad (9)$$

Under some relatively mild conditions on B and β, we could derive $\overline{p}_i$ in a simple closed form in the following theorem.

Theorem 1. (Steady-State Block Distribution) Let $B = \alpha_s m\lambda\overline{A}$ and $\beta = \frac{\alpha}{\alpha+\alpha_s}$. If β is a positive rational number, and $\frac{B}{\beta} \in \mathbb{Z}^{++}$ ($\mathbb{Z}^{++} = \{1, 2, \ldots\}$), then in the steady state, the fraction of segments with a total number of i blocks in users is given by,

$$\overline{p}_i = \binom{i + \frac{B}{\beta} - 1}{i}\beta^i(1-\beta)^{\frac{B}{\beta}}, \quad i = 0, 1, 2, \ldots. \quad (10)$$

In particular, the fraction of empty segments (segments which have no block among users) is given by $\overline{p}_0 = (1-\beta)^{\frac{B}{\beta}}$.

Remarks. In practice, the requirement that β is a rational number is naturally satisfied, since both the server number and the user number in the network are integers. Moreover, B is usually much larger than β and thus substituting B/β with the integer nearest to it will hardly affect the outcome of $\overline{p}_i$. Therefore, the value of $\overline{p}_i$ for nearly all valid β and B can be approximated by using the nearest β and B satisfying the conditions set in Theorem 1. Recall that we use $\boldsymbol{I}$ to denote the total number of blocks a segment has in users. Since the total number of segments G is large, the PMF of $\boldsymbol{I}$ will be asymptotically approximated by the values of $\overline{p}_i$, i.e., $\mathrm{P}\{\boldsymbol{I} = i\} = \overline{p}_i$. □

Proof. Let $C = B/\beta - 1$. We first prove $\overline{p}_i = \beta^i\binom{i+C}{i} \cdot \overline{p}_0$, for $i \geq 1$, and then determine $\overline{p}_0$ from the fact that $\sum_{i=0}^{\infty}\overline{p}_i = 1$. Let $\beta = \frac{l}{n} < 1$, where $l, n \in \mathbb{N}$. Because C is a non-negative integer, we have

$$\overline{p}_i = \overline{p}_0 \prod_{j=1}^{i}(\frac{l}{n} + \frac{B-\beta}{j})$$

By straightforward induction, we can finally get,

$$\overline{p}_i = \overline{p}_0(\frac{l}{n})^i \cdot \frac{(i+C)!}{i!C!} = \beta^i\binom{i+C}{i} \cdot \overline{p}_0$$

We now determine $\overline{p}_0$. Let $a_i = \overline{p}_i/\overline{p}_0 = \beta_i\binom{i+C}{i}$, $i = 0, 1, 2, \ldots$. Then we have $1/\overline{p}_0 = \sum_{i=0}^{\infty} a_i$. Hence, what we need to do is to determine $\sum_{i=0}^{\infty} a_i$. Because

$$\binom{i+C}{i} = \binom{C+i-1}{i} + \binom{C+i-1}{i-1}, \quad i \geq 1$$

we can get,

$$\sum_{i=0}^{\infty} a_i = \sum_{i=1}^{\infty}\beta^i\binom{(C-1)+i}{i} + \beta\sum_{i-1=0}^{\infty}\beta^{i-1}\binom{C+i-1}{i-1} + a_0$$

If we view a_i as a function of C and let $f(C) = \sum_{i=1}^{\infty} a_i(C) = \sum_{i=1}^{\infty}\beta^i\binom{i+C}{i}$, we can get the following iteration for $f(C)$,

$$\begin{cases} 0 = (\beta-1)f(C) + f(C-1) + \beta, & \text{for } C \geq 1 \\ f(0) = \sum_{i=1}^{\infty}\beta^i = \frac{\beta}{1-\beta} \end{cases}$$

By induction, it is not hard to get

$$f(C) = (1-\beta)^{-(C+1)} - 1, \quad C = 0, 1, 2, \ldots.$$

Hence,

$$\sum_{i=0}^{\infty} a_i = f(C) + a_0 = (1-\beta)^{-(C+1)}$$

Because $\sum_{i=0}^{\infty} \overline{p}_i = \overline{p}_0 \sum_{i=0}^{\infty} a_i = 1$, we finally get $\overline{p}_0 = (1-\beta)^{\frac{B}{\beta}}$. □

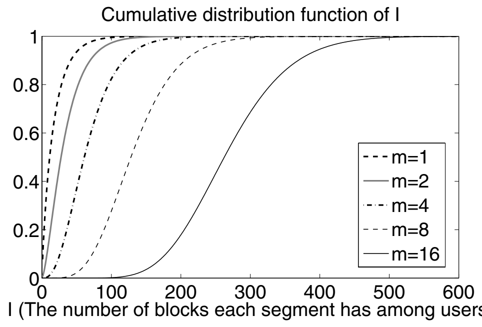

Fig. 2. Block distribution in the steady state under different segment sizes m (CDF of $\boldsymbol{I}$).

We have plotted the Cumulative Distribution Function (CDF) of $\boldsymbol{I}$ in Fig. 2 according to Theorem 1 under different segment sizes m (number of users $N = 1000$, number of servers $N_s = 50$, total number of blocks in the content $M = 1000$, upload rate $\lambda = \mu/k = 4$, average peer age $\overline{A} = 4$). The steady-state block distribution has a form of negative binomial distribution. It is not hard to show that such a distribution can be asymptotically approximated by a Gaussian distribution (refer to our technical report [21]). Apparently, as segment size m increases, there will be fewer rare blocks in the network, and thus the risk that the content becomes incomplete after server departures will be reduced. Another implication of Theorem 1 is that the block variation will be subdued as the segment size increases. We now quantitatively characterize the block variation in Theorem 2.

Theorem 2. (Steady-State Block Variation) Assume all the parameters satisfy the conditions set in Theorem 1. Let random variable $\boldsymbol{I}$ denote the total number of blocks a segment has in all the users, with the mean μ_I and variance σ_I^2. If we define block variation as $\gamma_I^2 = \frac{\sigma_I^2}{\mu_I^2}$, then in the steady-state,

$$\gamma_I^2 = \frac{1}{\alpha_s m \lambda \overline{A}} = \frac{F}{N_s m \mu \overline{A}}. \tag{11}$$

where $\overline{A}$ is the average age of a user given by Lemma 1.

Proof. We can derive the mean and variance of $\boldsymbol{I}$ from Theorem 1 through straightforward manipulation [21]:

$$\mu_I = \frac{B}{1-\beta}, \quad \text{and } \sigma_I^2 = \frac{B}{(1-\beta)^2}. \tag{12}$$

where $B = \alpha_s m \lambda \overline{A}$, $\beta = \frac{\alpha}{\alpha + \alpha_s}$. Thus, $\gamma_I^2 = \frac{\sigma_I^2}{\mu_I^2} = \frac{1}{\alpha_s m \lambda \overline{A}}$. □

We now discuss the important insights behind Theorem 2:

- The steady-state block variation is inversely proportional to the segment size m. Therefore, there can be a sweet spot in the curve of resilience-complexity tradeoff, which suffices to yield the major benefit of network coding.
- Network coding's true benefit lies in its resilience to block losses due to churn. When the average peer age $\overline{A}$ is small, the system can combat peer churn simply by increasing the segment size m for network coding. On the other hand, when peers all have long lifetimes, the problem of rare blocks is not salient and the block distribution could be balanced enough even without network coding.
- It is quite counter-intuitive that the block size k does not affect block variation at all. One might believe as block size k is increased, the total number of blocks in the content is reduced, and thus the imbalance among block distributions may be subdued. However, as k increases, blocks will be disseminated at a slower rate given limited bandwidth. Both effects counteract with each other, resulting in no change in block variation.
- The content size F and the number of online servers N_s are also critical parameters that affect block availability. The lack of servers will adversely affect block availability, not because servers hold more blocks than users, but because the upload behaviour of servers are fundamentally different from that of users, as we have pointed out in Sec. IV.

B. Content Availability in the Absence of Servers

In real-world P2P systems, servers may all become absent sometimes. In these cases, the content of interest suffers from a high risk of becoming incomplete. Suppose that all the servers leave the network at some point in steady state. In the absence of servers, it can no longer be guaranteed that any segment always remains decodable in the network. If the block number of certain segment falls below the segment size m, the content will become incomplete henceforth. On the other hand, if the content can be kept available for a sufficiently long time merely by users, then it is possible for the system to evolve into a new equilibrium again before the content turns incomplete, as some users may become servers later. Therefore, we ask the following questions: (1) *How long can the content be kept available, merely by the unreliable users?* and (2) *How much fraction of the content may still remain decodable at a given time after the servers have departed?* These metrics are important in that they represent the network's tolerance to the absence of servers with different segment sizes m for network coding.

First, it is worth noting that a necessary condition for the system to be stable is that $\alpha_s \neq 0$ ($N_s \neq 0$). An intuitive explanation for this fact is that with servers available, even if certain segments become unavailable among users, servers could still replenish blocks of these segments to the network. In contrast, without servers, i.e., when $\alpha_s = 0$ ($N_s = 0$), a segment becomes unavailable forever once its block number reduces to less than m, with no possibility to be replenished again. Actually, we have solved for $\overline{p}_i$ by letting $\frac{\mathrm{d}s_i}{\mathrm{d}t} = 0$ in the case of $\alpha_s = 0$, and found $\overline{p}_i = 0$ for any finite i in steady state. According to Fig. 1, this indicates an extreme case that nearly all the segments will finally have zero blocks, while only one segment will have an infinite number of blocks in the network. Thus, the system can not hold stable in the absence of servers.

We assume that the servers leave the network altogether in the steady state and set time $t = 0$ on the servers' departure. We then solve the differential equations to derive parameters of interest. Since servers leave in the steady state, we obtain initial conditions at time $t = 0$:

$$p_i(0) = \overline{p}_i = \binom{i + \frac{B}{\beta} - 1}{i} \beta^i (1-\beta)^{\frac{B}{\beta}}, \quad i = 0, 1, 2, \ldots, \tag{13}$$

where $B = \alpha_s m \lambda \overline{A}$, $\beta = \frac{\alpha}{\alpha + \alpha_s}$, β is a positive and rational number and $\frac{B}{\beta} \in \mathbb{Z}^{++}$. By letting $N_s = 0$, i.e., $\alpha_s = 0$, we obtain the differential equations for the system after the servers' departure:

$$\begin{aligned} \overline{A} \cdot \frac{\mathrm{d}s_i(t)}{\mathrm{d}t} &= (i-1)p_{i-1}(t) - i p_i(t), \quad \text{for } i \geq 1 \\ s_0(t) &= 1. \end{aligned} \tag{14}$$

Hence, the problem of determining the content loss at a given time has been converted to the problem of solving the system of ODEs (14) subject to the initial conditions (13).

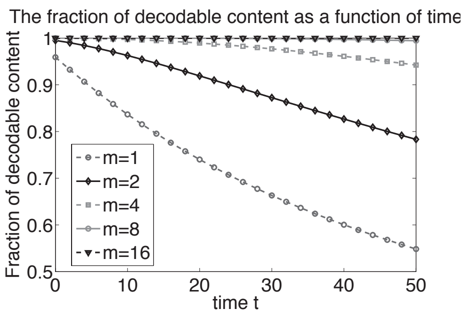

Fig. 3. The fraction of decodable content at time t after server departures under different segment sizes m.

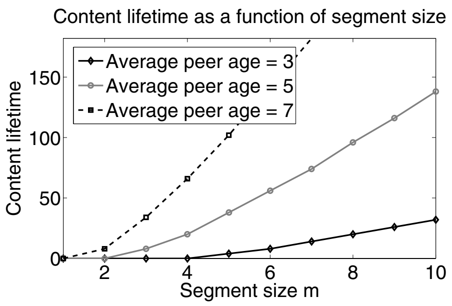

Fig. 4. Content lifetimes after server departures under different segment sizes m and different average peer age $\overline{A}$.

When the segment size is m, it is not hard to figure out that the percentage of content loss at time t is the fraction of segments with fewer than m blocks in the network, which is $1 - s_m(t)$. Hence, as we vary the segment size m, two factors will intertwine to affect the content loss fraction. The first factor is the initial condition at $t = 0$ which depends on m; as the segment size increases, either less content will be lost right upon server departures, or segments are distributed in a more balanced way if the content is still complete. The second factor is the objective function $1 - s_m(t)$ that stands for the content loss percentage at time t. We have plotted the fraction of decodable content as a function of time t after server departures in Fig. 3 under different segment sizes (number of users $N = 1000$, number of servers $N_s = 50$, total number of blocks in the content $M = 1000$, upload rate $\lambda = \mu/k = 4$, average peer age $\overline{A} = 5$). It is clearly shown that, as the segment size m increases, not only does the system lose less content right upon the servers' departure, but the content loss rate after the servers' departure is reduced greatly as well. Besides, merely a small increment in m would result in a salient decrease in the content loss rate.

Finally, we numerically determine content lifetime in a similar way. The content lifetime is defined as the period from server departures to the incompleteness of the first segment. Assume that the segment size is m, and the total number of segments is $G = M/m$. If all the segments are still decodable upon server departures, then the content becomes incomplete when there is at least one segment which has less than m blocks in the network. It is equivalent to saying that the content becomes unavailable at the time when the fraction of segments with less than m blocks grows to greater than $\frac{1}{G}$. Thus, the content lifetime equals to the first hitting time of the function $1 - s_m(t)$ to $\frac{1}{G}$ starting from the initial conditions set by (13). The result has been plotted in Fig. 4, where the parameters are set to the same value as in the previous figure. Similarly, an increasing trend in content lifetime is observed as the segment size increases.

VI. Experimental Results

We have developed a simulating environment using C++ to experimentally evaluate the behaviour of dynamic P2P networks. In most experiments, more than 2,000,000 peers are involved in total, with 6,000 peers being simultaneously online at any given time.

Let us first briefly describe the settings of our simulator (please refer to our technical report [21] for a complete description). There exists a tracking server in the system, which helps maintain the neighborhood of a peer. However, all other tasks, including downstream peer selection and data exchanges, are performed locally at peers. Each peer is connected to at least 40 other peers (in accordance to BitTorrent), which form its neighborhood. We fix the number of online peers while allowing peer churn: a peer will leave the network when its assigned lifetime has expired and a new peer will join at the same time. To join the session, a new peer should contact the tracking server, which in response gives it 40 randomly chosen online peers to connect to.

The simulator is run in time slots (rounds). In each time slot, a peer first randomly chooses 4 (as in BitTorrent) of its neighbors to serve in this slot. This process is called downstream peer selection. However, under such a scheme, there may exist peers not chosen by any other peers in a round, whereas in the meantime other peers may have a large number of upstream peers. To avoid such a load imbalance, we require each peer to have at most 6 upstream peers. With this policy, most peers will have at least one upstream peer in each round. Afterwards, each peer will transmit data to its downstream peers using group network coding as mentioned in Sec. III. The upload bandwidth of a peer is measured in Kilobytes (KB) per round.

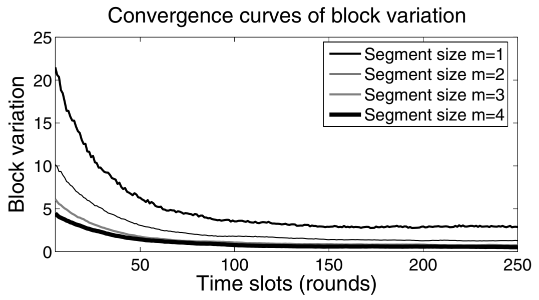

Fig. 5. The convergence of the block variation.

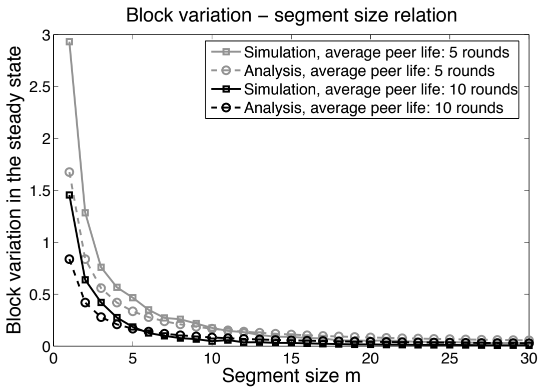

Fig. 6. Steady-state block variation as a function of segment size m. Average lifetime of user peers $\overline{L} = 5$ rounds.

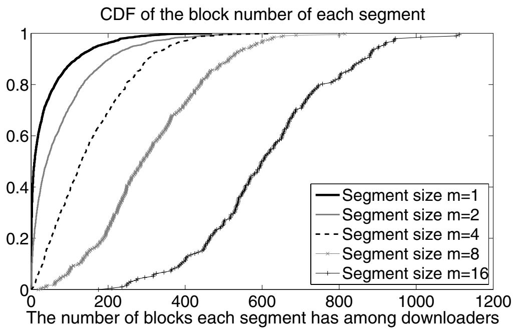

Fig. 7. Steady-state CDF of the number of blocks each segment has among users Average lifetime of user peers $\overline{L} = 5$ rounds.

First, it is clearly shown in Fig. 5 that the system does have a steady state in which block distribution stays roughly constant. Parameters are set such that the number of online user peers $N = 6000$, number of online servers $N_s = 100$, file size $F = 768$ MB, block size $k = 256$ KB. Each user has a lifetime exponentially distributed with a mean of 5 rounds. To accommodate heterogeneity, we assume an equal number of peers with high upload bandwidth (2 MB/round), medium upload bandwidth (512 KB/round), and low upload bandwidth (256 KB/round) connectivity.

Fig. 6 shows the block variation γ_I^2 in steady state. The experiment ran for 500 rounds. And after round 200, the system stepped into the steady state. Parameters are set to the same values as in the previous experiment. It clearly demonstrates our main theoretical result: γ_I^2 is inversely proportional to the segment size m. However, we do not fully understand why the simulation and the analysis are different from each other by a constant coefficient. This is probably because we have adopted a time-synchronized model in the simulation, whereas in the analysis we have not. To acquire a sense of how the blocks of each segment are distributed in steady state, the block distribution in the same experiment is plotted in Fig. 7, which also substantiates the correctness of Theorem 1.

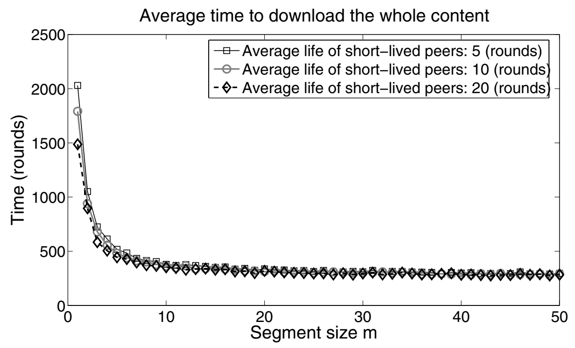

Fig. 8. The average time for a long-lived peer to download the entire content in steady state under different segment sizes m. The average lifetime of short-lived peers is shown in the figure.

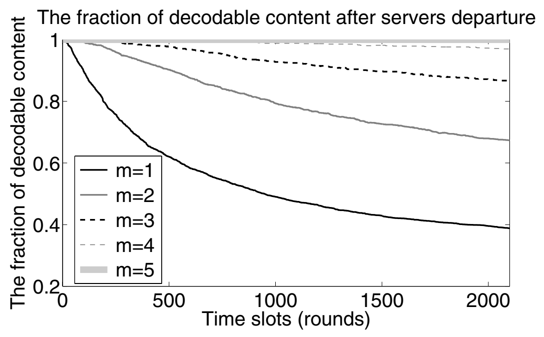

Fig. 9. The fraction of decodable content after server departures as time passes.

We now link the average download time experienced at a long-lived user peer with the block variation and segment size m. The results are plotted in Fig. 8. Parameters are set such that the number of online user peers $N = 2000$, the number of online server peers $N_s = 30$, file size $F = 256$ MB. Bandwidths and block size are the same as in the previous experiments. 1% of all the user peers are long-lived ones which will not leave until they finish downloading. The other peers are short-lived ones which have lifetimes exponentially distributed. It is clearly shown in Fig. 8 that as segment size increases, the average download time required to obtain the entire content is reduced. And the relation between download time and segment size assumes a similar form (inverse proportion) to that between block variation and segment size. The underlying reason is that as block variation is subdued, peers will not be hindered in obtaining those rare blocks. Besides, there is a sweet spot of segment size, beyond which the download time can hardly be further reduced. Thus, the use of a small segment size, such as 10-20, suffices to optimize the download efficiency in the system, while only incurring a moderate computational complexity.

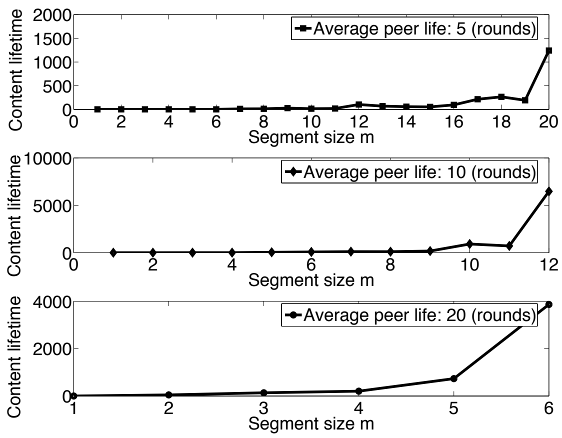

Fig. 10. The relationship between content lifetime and segment size m.

We have also plotted the fraction of decodable content as a function of time after servers' departure in Fig. 9. Servers leave in the steady state (round 400), and we set $t = 0$ upon servers' departure. There are 6000 users and 100 servers being simultaneously online, sharing a file of size 768 MB. The average lifetime of user peers is $\overline{L} = 20$ (rounds). Other parameters are the same as those of the previous experiment. We could see that the results agree with the analysis in trend. Finally, under this setting, we show content lifetimes after servers' departure for different segment sizes and different average peer lifetimes in Fig. 10. Two conclusions can be drawn from Fig. 10. First, content lifetime increases in trend as segment size increases. Second, when peers have longer lifetimes, the effect of churn becomes less severe, and thus content lifetime is significantly increased.

VII. Concluding Remarks

In this paper, we study the fundamental resilience-complexity tradeoff of applying network coding in P2P content distribution sessions with severe peer dynamics. Our major finding is that the true benefit of network coding in P2P networks lies in its resilience to peer churn, which is an important issue not well understood in the existing literature. We have theoretically proved that as peers abort frequently, blocks will be taken away randomly and some blocks may become rare, adversely affecting download efficiency. With network coding, all the blocks within a segment are equally useful, and thus the problem of significant degrees of imbalance with respect to block availability, namely, block variation, is addressed. By varying the segment size for network coding, we quantified such metrics as block variation, download times, content loss rate and content lifetime under different coding complexities. We have discovered that the block variation is inversely proportional to the segment size. A similar relation between the download time and segment size is also observed. Therefore, small segment sizes — less than 20 in our simulations even with high peer volatility — suffice to realize the major benefit of network coding in terms of reducing download times and enhancing content availability.

References

[1] R. Ahlswede, N. Cai, S. R. Li, and R. W. Yeung, "Network Information Flow," *IEEE Transactions on Information Theory*, vol. 46, no. 4, pp. 1204–1216, July 2000.

[2] S. R. Li, R. W. Yeung, and N. Cai, "Linear Network Coding," *IEEE Transactions on Information Theory*, vol. 49, pp. 371, 2003.

[3] R. Koetter and M. Medard, "An Algebraic Approach to Network Coding," *IEEE/ACM Transactions on Networking*, vol. 11, no. 5, pp. 782–795, October 2003.

[4] C. Gkantsidis and P. Rodriguez, "Network Coding for Large Scale Content Distribution," in *Proceedings of IEEE INFOCOM 2005*, March 2005.

[5] M. Wang and B. Li, "How Practical is Network Coding?," in *Proc. of 14th IEEE International Workshop on Quality of Service (IWQoS'06)*, New Haven, CT, USA, June 2006.

[6] P. Chou, Y. Wu, and K. Jain, "Practical Network Coding," in *Proceedings of Allerton Conference on Communication, Control and Computing*, October 2003.

[7] T. Ho, M. Medard, J. Shi, M. Effros, and D. Karger, "On Randomized Network Coding," in *Proceedings of Allerton Conference on Communication, Control and Computing*, October 2003.

[8] C. Gkantsidis, J. Miller, and P. Rodriguez, "Comprehensive View of a Live Network Coding P2P System," in *Internet Measurement Conference (IMC 2006)*, 2006.

[9] Y. Wu, "Network Coding for Multicasting," Tech. Rep., Princeton University, Nov. 2005.

[10] D. S. Lun, M. Medard, R. Koetter, and M. Effros, "Further Results on Coding for Reliable Communication Over Packet Networks," in *Proc. of IEEE International Symposium on Information Theory (ISIT2005)*, Sept. 2005.

[11] S. Acedanski, S. Deb, M. Medard, and R. Koetter, "How Good is Random Linear Coding based Distributed Networked Storage?," in *First Workshop on Network Coding, Theory, and Applications (NetCod'05)*, Riva del Garda, Italy, April 2005.

[12] B. Fan, D. M. Chiu, and J. C. S. Lui, "Stochastic Analysis and File Availability Enhancement for BT-like File Sharing Systems," in *Proc. of 14th IEEE International Workshop on Quality of Service (IWQoS'06)*, New Haven, CT, USA, June 2006.

[13] B. Cohen, "Incentives Build Robustness in BitTorrent," in *P2P Economics Workshop*, Berkeley, CA, USA, 2003.

[14] X. Yang and G. de Veciana, "Service Capacity of Peer to Peer Networks," in *Proceedings of IEEE INFOCOM 2004*, Hong Kong, China, March 2004.

[15] D. Leonard, V. Rai, and D. Loguinov, "On Lifetime-Based Node Failure and Stochastic Resilience of Decentralized Peer-to-Peer Networks," in *Proc. of ACM SIGMETRICS'05*, Banff, Alberta, Canada, June 2005.

[16] M. Mitzenmacher, "Load Balancing and Density Dependent Jump Markov Processes," in *Proc. of the 37th Annual IEEE Symposium on Foundations of Computer Science (FOCS 96')*, 1996, pp. 213–223.

[17] K. K. Ramakrishnan and R. Jain, "A binary feedback scheme for congestion avoidance in computer networks," *ACM Transactions on Computer Systems*, vol. 8, no. 2, pp. 158–181, May 1990.

[18] T. G. Kurtz, "Approximation of Population Processes," in *CBMS-NSF Regional Conference Series in Applied Math.* SIAM, 1981.

[19] M. Mitzenmacher, "Studying Balanced Allocations with Differential Equations," in *Technical Note 1997-024*, Palo Alto, CA, October 1997, Digital Equipment Corporation Systems Research Center.

[20] G. Tan and S. A. Jarvis, "On the Reliability of DHT-Based Multicast," Tech. Rep., University of Warwick, 2006.

[21] Di Niu and Baochun Li, "Quantifying the Resilience-Complexity Tradeoff of Network Coding in Dynamic P2P Networks," Tech. Rep., http://iqua.ece.toronto.edu/papers/ResilienceNC.pdf, ECE, University of Toronto, February 2007.

[22] S. Deb, M. Medard, and C. Choute, "Algebraic Gossip: A Network Coding Approach to Optimal Multiple Rumor Mongering," *IEEE Transactions on Information Theory*, vol. 52, no. 6, June 2006.