On optimizing for epidemic live streaming

Nidhi Hegde, Fabien Mathieu, Diego Perino
Orange Labs, France, first.last@orange-ftgroup.com

Abstract—Optimal dissemination schemes have previously been studied for peer-to-peer live streaming applications. Live streaming being a delay-sensitive application, fine tuning of dissemination parameters is crucial. In this paper, we investigate optimal sizing of chunks, the units of data exchange, and probe sets, the number peers a given node probes before transmitting chunks. Chunk size can have significant impact on diffusion rate (chunk miss ratio), diffusion delay, and overhead. The size of the probe set can also affect these metrics, primarily through the choices available for chunk dissemination. We perform extensive simulations on the so-called random-peer, latest-useful dissemination scheme. Our results show that size does matter, with the optimal size being not too small in both cases.

I. Introduction

Peer-to-peer data transfer has been a dominant source of network traffic for the past few years. Peer-to-peer mechanisms of transfer that rely on client uploads have also been used recently for live video streaming solutions such as PPLive [3], CoolStreaming [11]. Natural questions that arise for live video streaming concern whether the delay and quality requirements can be met by distributed client-based dissemination.

In most cases, a P2P live streaming algorithm splits the stream into chunks (also termed pieces). Chunks are considered as the atomic components of the stream, and a peer can only send chunks it has fully received. Much of the work in the literature has been devoted to the search for a chunk exchange policy that is feasible and optimal. We focus here on the unstructured epidemic approaches [7], [1], [8], where the policy is described by a *scheme* that indicates which chunk a given peer should try to send, and to whom.

A good scheme is indeed essential to the epidemic live streaming problem. For a given scheme however, an optimization at a detailed level is also important. This involves the fine tuning of dissemination parameters, such as chunk size, receiver buffer size, number of peers to probe, etc. The chunk size has a significant impact on performance, since smaller chunk sizes may be more efficient but incur relatively higher overhead, and larger chunk sizes have lower overhead but may result in higher delay. The receiver buffer size (relative to chunk size) impacts the diversity in choice available to a peer for transmission. In the scheme with random peer choice, probing more than one peer for the decision of chunk exchange may help (power of choices), but it also increases overhead. These are some of the finer details of any dissemination scheme that must be closely examined.

There has been some study on parameter sizing for peer-to-peer file sharing systems. In [6] it is shown that small chunk

sizes are not always best for file transfer; [4] proposes uplink allocation strategies designed to improve uplink utilization of BitTorrent-like systems. However, results obtained for file sharing systems are not directly applicable to live streaming applications. First, a newly created chunk should be disseminated as fast as possible in live streaming, so there is a strong delay component, naturally limiting the chunk size. Secondly, missing chunks may be acceptable if a resilient codec is used, so optimal values are not always comparable to those in the file transfer case. Then, the buffer size, which is a parameter specific to streaming, can impact the performance (see for instance [12]).

In this paper, we investigate dissemination parameters in peer-to-peer live video streaming through extensive simulations. Specifically, we focus on the *rp/lu* diffusion scheme, where a peer sends the latest (freshest) useful chunk to a randomly selected peer. We will also briefly consider other schemes for comparison. We will show that indeed chunk size significantly impacts the performance. In fact, there is a range of chunk sizes that may be suitable, where the specific choice of the chunk size ultimately depends on the delay/chunk miss ratio trade-off. We will also show that a fine tuning of the number of peers to probe and the number of simultaneous chunks to send is important.

The rest of the paper is organized as follows. In the next Section we outline our simulation framework. Section III covers the impact of the chunk size, and highlights the suitable range of chunk sizes among various dissemination schemes. Section IV examines the value of the number of peers to probe for chunk dissemination. We finally conclude the paper in Section V.

II. Methodology

Epidemic diffusion schemes and their behavior have been extensively studied in the literature. See for instance [1] for a detailed study and references therein. Here, we consider schemes where the sender first selects the destination peer and then sends a chunk. We focus on this particular class of schemes because they are fairly simple and thus allow us to focus on the impact of the parameters such as chunk size. Moreover, they are efficient in terms of diffusion rate and delay, and can potentially generate low overhead.

In particular we consider three algorithms representative of this class:

- *random peer / latest blind chunk* (*rp/lb*). The destination peer is chosen uniformly at random among sender peers' neighbors and the most recent chunk in the buffer is

selected (regardless of whether the receiver needs that chunk or not);

- *random peer / latest useful chunk (rp/lu)*. The destination peer is chosen uniformly at random among sender peers' neighbors and the most recent chunk not own by the receiver peer is selected; unless otherwise specified, this is the scheme considered in this paper.

- *bandwidth aware peer / latest useful chunk (ba/lu)*. This scheme is inspired by [2]. A peer i is selected with a probability proportional to its upload bandwidth u_i and the most recent chunk not own by the receiver peer is selected. Note that for homogeneous upload bandwidths this is equivalent to *rp/lu*.

In order to analyze these algorithms under the same framework and derive general results, we used an event-based simulator developed by the Telecommunication Networks Group of Politecnico di Torino[1]. The simulator has been modified to take network latencies, control overhead and parallel upload connections into account.

In our simulator we assume that the overlay network is an Erdös-Renyi graph $\mathcal{G}(n, p)$, where n is the size of the peer population and p is the probability that a link connecting two peers does exist. Every peer i has therefore a partial view of the overlay network, with an average number of neighbors $p(n-1)$. The choice of this random graph model is to avoid results influenced by specific overlay structures.

We assume that every link connecting a pair of peers $\{i, j\}$ is characterized by a constant round trip delay RTT_{ij} and is lossless. We further assume that there are no queuing nor processing delays, so the *transfer delay* (the time for a chunk or control packet to travel from peer i to peer j) is equal to $transmission\ delay + \frac{RTT_{ij}}{2}$. The choice of such a network model allows us to obtain results that are not affected by the overlay network structure or by transport network congestion or losses. A peer is characterized by its upload bandwidth u_i. There is a single source S with upload capacity u_s and a limited overlay knowledge as well.

Every peer periodically selects a subset m of its neighbors, according to one of the aforementioned algorithms (that is random or bandwidth-aware selection), and probes them in order to discover their missing chunks, except for the case of the *latest blind* scheme. We refer to the set of neighbors probed as the probe set. Based on the responses possibly received, the peer then transmits corresponding chunks.

A peer can upload a chunk to at maximum m' peers in parallel by fairly sharing its upload bandwidth. It may happen that a peer cannot serve m' recipients because it does not have enough useful chunks. In that case it uploads the chunks faster (since there are less than m' active connections), but it may stay idle for the subsequent period of time (because it needs to acquire new chunk maps from newly selected peers). An additional overhead is taken into account at every peer to reply to control messages coming from potential sender peers.

Unless otherwise stated we consider a network of $n = 1000$

peers, all with the same upload bandwidth $u_i = 1.03 Mb/s$, an unlimited download bandwidth and about 50 neighbors ($p = 0.05$). We set the stream rate $s = 0.9 Mb/s$. This setting represents a system with a bandwidth over-provisioning of 14% in order to avoid results influenced by *critical regime* effects ($u_i = SR$ see [1]), or by a too restricted neighborhood. Latencies between nodes are taken from the data set of the Meridian project [9]. A buffer of size up to 300 chunks is available at all peers, in order to avoid possible missing chunks due to buffer shortage (this implies a buffer size proportional to the chunk size).

III. Chunk Size and performance

When considering a streaming algorithm, a crucial performance metric is the diffusion rate/diffusion delay/overhead trade-off achieved by that algorithm, which can be summarized by a (*chunk miss ratio*,*delay*,*overhead*) triplet.

Following [5], we define the *chunk miss ratio* as the asymptotic probability to miss a chunk (or equivalently the difference between the stream rate s and the actual goodput), while *average diffusion delay* is defined as the delay between the creation of a chunk and its reception by a peer, averaged over the successful chunk transmissions. Note that since links are lossless, a peer misses a given chunk only if none of its neighbors has scheduled that chunk for it. The *overhead* is defined as the difference between the bandwidth used by peers (throughput) and the actual data received (goodput). In our framework, this overhead is due only to control messages exchanged between peers.

As a first experiment, we analyze the performance triplet as a function of the chunk size. The results are shown in Figures 1 to 3, for the *rp/lu* scheme with $m = m'$ varying from 1 to 5.

A. Chunk miss ratio

In Figure 1, we observe two cases:

- For large chunks (in our experiment, c greater than a few hundred kilobits, the exact value depends on the number of simultaneous connections m), there are no missing chunks.

- As the chunk size goes below a certain critical value, chunks start to miss, roughly proportional to the logarithm of the chunk size.

This phenomenon can be explained as follows: the time between two consecutive chunks is c/s, and is therefore proportional to the chunk size c. When c is big enough (all other parameters being the same), we can assume that more and more control messages per chunk can be exchanged between peers. This should achieve a proper diffusion, provided enough bandwidth is available, since a sender peer will have enough time to find a neighbor needing a given chunk. On the contrary, when c/s is too small, peers do not have enough time to exchange control messages, resulting in missing chunks. Note that increasing m slightly improves the performance.
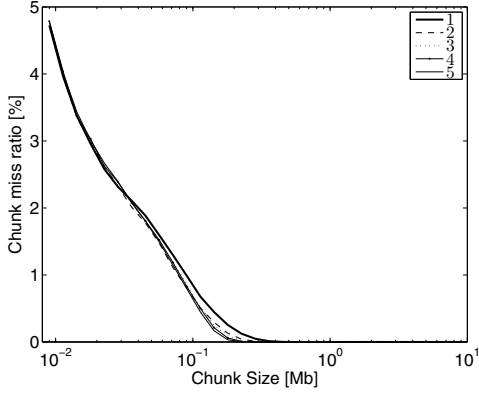
Fig. 1. Chunk miss ratio as a function of the chunk size. $m = m'$ varying from 1 to 5.

Fig. 2. Average diffusion delay as a function of the chunk size

B. Delay

The average diffusion delay as a function of the chunk size is shown in Figure 2. The main result is that the delay is proportional to the chunk size (hence the linear x-axis used; although difficult to observe on the figure, the proportional relationship was also verified for small values.). We also note that it grows with m.

In fact, the chunk is the unit of data exchange and a peer can re-transmit a chunk only if it has fully received it. To increase the chunk size increases the time needed to exchange a chunk, and as a consequence the diffusion delay increases. To increase the number of parallel upload connections, leads to a similar effect: the time needed to exchange a chunk increases with the number of connections because the upload bandwidth is shared with more nodes, and as a consequence the diffusion delay increases.

This result is consistent with theoretical results obtained in [8] where RTT is neglected and the chunk transmission time is simply considered inversely proportional to the sender's bandwidth. Under that framework, the minimal diffusion delay is given by:

$$d_{\min} = \frac{mc\ln(n)}{\ln(1+m)s}. \tag{1}$$

C. Overhead

The performance with respect to overhead, i.e. the difference between the throughput and goodput, is shown in Figure 3 (only the curves for $m = 1$ and $m = 5$ are displayed for legibility). For very small chunks, we have a non-intuitive trend, where as c grows, the goodput increases *and* the throughput decreases (or equivalently, the overhead decreases faster than the goodput increases). This process slows down so that at some point the throughput increases again. For big enough chunks, the overhead becomes roughly constant (for a given m), while the goodput becomes equal to the stream rate (meaning no missing chunks).

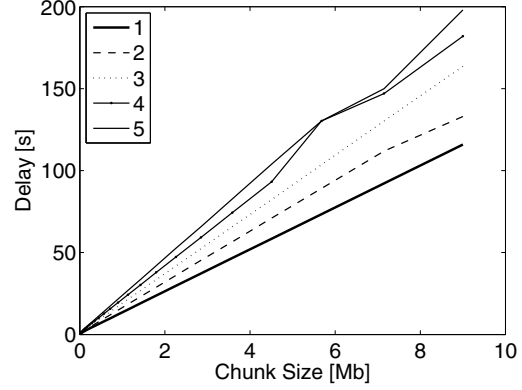The goal of this paper is not to give a complete explanation of the observed results, but rather give some intuitions behind the overhead behavior. For very small chunks, chunk miss ratio is high, which, as mentioned earlier, come from the fact that not enough control messages can be sent. Asymptotically, we may imagine that only one control message per sent chunk is produced, resulting in an overhead/goodput ratio of $\frac{c_c}{c}$, where c_c is the size of a control message.

On the other hand, in the limit as the chunk size is increased, we may expect that a peer can send a number of messages per sent chunk that is proportional to the chunk characteristic time c/s. This would result in an overhead ratio proportional to $\frac{c_c}{s}$, and thus independent of c (but not of other parameters like the median RTT or m).
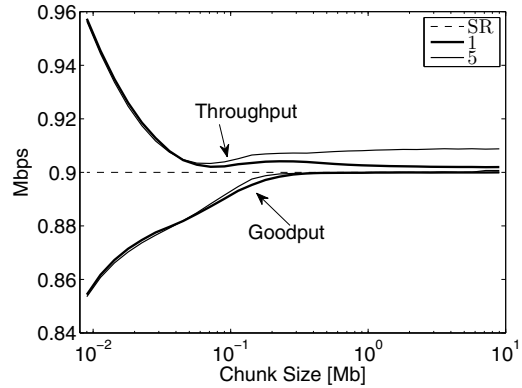

Fig. 3. Goodput and throughput as a function of the chunk size, the overhead being the difference. The stream rate s is also indicated.

D. Suitable range for c

In light of the study above, there is a good order of magnitude for suitable chunk size in epidemic live streaming. For the parameters considered here, c should be greater than 0.06 Mb (which corresponds to about 15 chunks per second) and smaller than 0.3 Mb (3 chunks per second):

- to send the stream at more than 15 chunks per second is good for the delay (which stays roughly proportional to c), but results in both an increase in throughput and a decrease in goodput;

- goodput and throughput are stationary for c greater than 0.3 Mb: using bigger chunks only means longer delay;
- between these values, the choice of c results in a chunk miss ratio/delay trade-off: smaller delay with some missing chunks or greater delay with no missing chunks. Choosing a precise value for c depends then on factors that will not be discussed here, such as the codec used, the required QoS, etc.

In our experiments the suitable range for chunk size begins when the chunk characteristic time ($\frac{c}{s}$) has the same order of magnitude than the median RTT, and ends an order of magnitude later. We scaled the RTT distribution used in order to observe the evolution of the range with the median RTT. The results, reported in Figure 4, show that the range values are indeed roughly proportional to the median RTT.

Note that the lower bound of the suitable range gives an indication on the minimal delay that can be achieved without too much missing chunks and overhead. In section IV, we will see that enhanced diffusion techniques can help lower that bound.
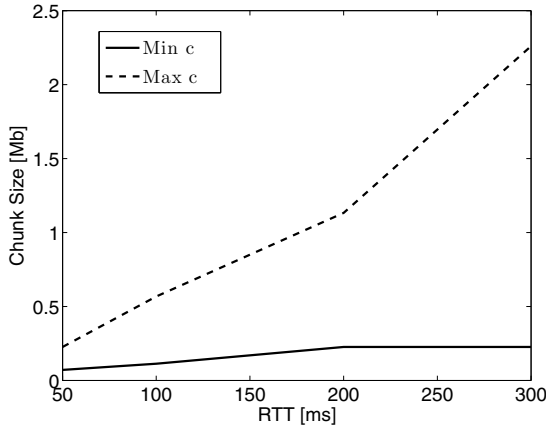

Fig. 4.   Suitable range (for $m' = m$)

We have performed experiments using various diffusion schemes, RTT and bandwidth distributions, values of probe set m, stream rate s and so on. All results are not report here for lack of space but, even if given metric values may differ, we observed the existence of a *suitable* range for c.

As an example, in the following we compare the suitable range of chunk size for two RTT values and the three following dissemination schemes: *rp/lu*, *rp/lb* and *ba/lu*. Since the scenarios with homogeneous bandwidths are identical under the *rp/lu* and *ba/lu* schemes, we use a heterogeneous bandwidth distribution derived from [10]. We set $m = m' = 1$, and we plot the throughput, goodput and average delay for these cases using two values of latency, RTT = 50, 100 ms (Figure 5).

Note that the scheme *rp/lb* suffers high chunk miss ratios for all values of chunk sizes considered. Indeed it has been shown [1] that this scheme performs poorly with respect to rate, while being optimal with respect to delay. The scheme *rp/lu* has fewer missing chunks, but higher delay, while the

performance of *ba/lu* lies between the other schemes for both chunk miss ratio and delay.

However, beyond the fact that the chunk miss ratio/delay/overhead trade-off is closely related to the scheme (more complete studies are available elsewhere [1], [8], [2]), the striking observation is that all these schemes admit a similar suitable range for c, which seems to scale with the median RTT of the network. This supports our claim that the suitable range for c depends mainly on the median RTT and s (the inter-chunk delay $\frac{c}{s}$ should have roughly the same order than the RTT), the actual scheme being secondary.
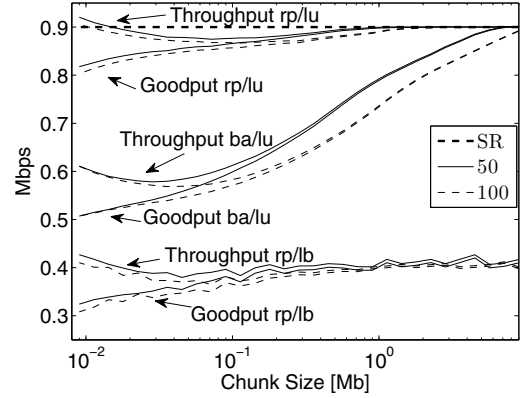
(a) Average goodput and throughput
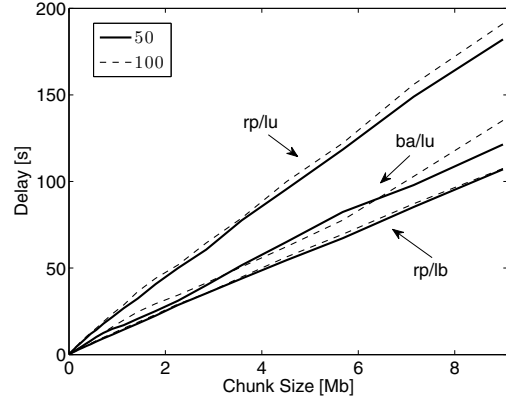
(b) Average diffusion delay

Fig. 5.   *rp/lb*, *rp/lu* and *ba/lu* comparison

IV. SIZE OF PROBE SET

In the results presented so far, we have assumed that the number of simultaneous exchange chunks, m', is identical to the size of the probe set m. We now consider the impact of probing more peers than the number of simultaneous chunks sent. A larger probe set affords a sender peer a higher chance to find a recipient peer for whom it has useful chunks (power of choices principle). However, it also increases overhead, and possibly delay.

Figure 6(a) plots the chunk miss ratio/delay trade-off for various m'/m pairs. The scheme is *rp/lu*, the bandwidth is

homogeneous and the chunk size is set to $c = 0.15$ Mb (middle of the suitable range). The figure shows that using $m' = m$ is not optimal, and having a larger probe set, $m > m'$ significantly reduces both delay and missing chunks. The delay decreases from about 10 s for the $m = m'$ case, to less than 4 s for the $1/3, \ldots, 6$ cases (meaning $m' = 1$ and $m = 3, \ldots, 6$). With regards to the chunk miss ratio, there are some (m'/m) pairs for which no missing chunks could be observed in our experiment: $1/3 - 6$, $2/5 - 6$, $3/5 - 6$, $4/6$. This suggests that a consequence of using $m' < m$ is a shift of the suitable range for c.
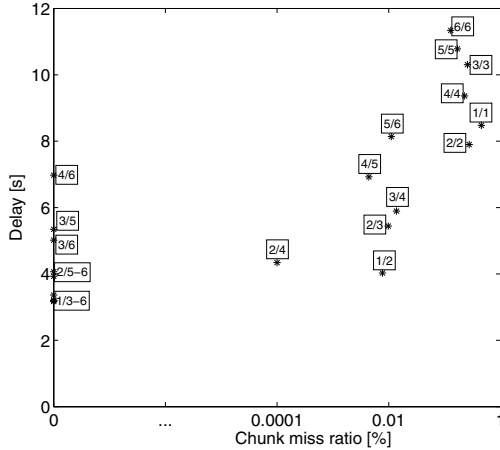
(a) $c = 0.15$ Mb (middle of the suitable range for $m = m'$)
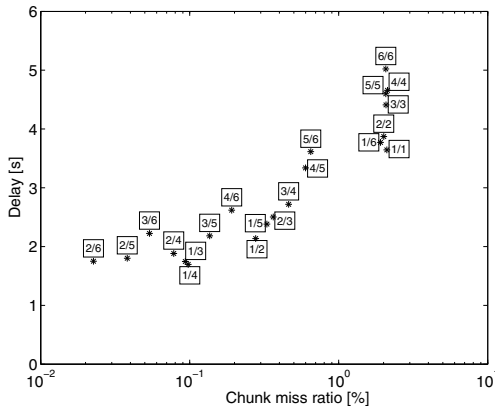
(b) $c = 0.035$ Mb (below the suitable range for $m = m'$)

Fig. 6. m'/m chunk miss ratio/delay trade-off for two values of c

In order to verify this interpretation, we now set $c = 0.035$ Mb, which is clearly below the suitable range observed in § III-D for $m = m'$. The results are shown in Figure 6(b).

We observe that no pair (m'/m) can achieve diffusion without missing chunks for such a small c, however the trade-offs are still worthwhile with respect to the delay: using $m'/m = 2/6$, we get a delay of 1.7 s with a chunk miss ratio of about 0.02 %, which represents an excellent trade-off, far better than the one observed for $c = 0.15$Mb and $m = m'$. This indicates that $c = 0.035$ Mb is definitively within the suitable range for $m'/m = 2/6$.

Also note how the relative efficiency of the various m'/m values is impacted by the choice of c: for instance, $1/6$, which is optimal for $c = 0.15$ Mb, performs rather poorly for $c = 0.035$ Mb. Although the results presented here refer to the *rp/lu* scheme, we performed experiments with other schemes and we observed similar trends, confirming that using a proper $m' < m$ can significantly improve the delay.

On the other hand, there is a price for going below the suitable range defined in § III-D: for a given scheme, the overhead still depends on m and c. For *rp/lu*, it stays close to the overhead displayed in Figure 3 even for $m' < m$. So using small c with $m' < m$ can reduce the delay, but it requires more throughput.

V. Conclusion

We have investigated the dissemination parameters of peer-to-peer epidemic live video streaming through extensive simulations. We have shown that the chunk size significantly impacts performance and that the chunk size should fall within a given range which is mostly determined by the median RTT of the network and the stream rate.

We have also shown that the size of the probe set affects performance of diffusion schemes, and, in particular, a probe set larger than the actual number of concurrent connections may improve miss ratio/delay performance by modifying the suitable chunk size ranges.

References

[1] T. Bonald, L. Massoulié, F. Mathieu, D. Perino, and A. Twigg. Epidemic live streaming: optimal performance trade-offs. In *Proc. of SIGMET-RICS*, 2008.
[2] A. P. C. da Silva, E. Leonardi, M. Mellia, and M. Meo. A bandwidth-aware scheduling strategy for p2p-tv systems. In *Proc. of P2P*, 2008.
[3] X. Hei, C. Liang, J. Liang, Y. Liu, and K. W. Ross. Insights into PPLive: A measurement study of a large-scale P2P IPTV system. In *Proc. of IPTV Workshop, WWW*, 2006.
[4] N. Laoutaris, D. Carra, and P. Michiardi. Uplink allocation beyond choke/unchoke or how to divide and conquer best. In *Proc. of CoNEXT*, 2008.
[5] C. Liang, Y. Guo, and Y. Liu. Is random scheduling sufficient in P2P video streaming? In *Proc. of ICDCS*, pages 53–60, Los Alamitos, CA, USA, 2008.
[6] P. Marciniak, N. Liogkas, A. Legout, and E. Kohler. Small is not always beautiful. In *Proc. of IPTPS*, 2008.
[7] L. Massoulié, A. Twigg, C. Gkantsidis, and P. Rodriguez. Randomized decentralized broadcasting algorithms. In *Proc. of INFOCOM*, 2007.
[8] F. Picconi and L. Massoulié. Is there a future for mesh-based live video streaming? In *Proc. of P2P*, 2008.
[9] M. Project. http://www.cs.cornell.edu/People/egs/meridian/.
[10] S. Saroiu, P. K. Gummadi, and S. Gribble. A measurement study of peer-to-peer file sharing systems. In *MMCN*, 2002. http://www.cs.washington.edu/homes/gribble/papers/mmcn.pdf.
[11] X. Zhang, J. Liu, B. Li, and T. Yum. Coolstreaming/donet : A data-driven overlay network for Peer-to-Peer live media streaming. In *Proc. of INFOCOM*, 2005.
[12] Y. Zhou, D.-M. Chiu, and J. C. S. Lui. A simple model for analyzing p2p streaming protocols. In *Proc. of ICNP*, 2007.