Pull-based Peer-to-Peer Streaming: Fundamental Limits & Performance Gap

Chen Feng, Baochun Li, and Bo Li
Dept. of Electrical and Computer Engineering
University of Toronto
Dept. of Computer Science
Hong Kong University of Science & Technology
Success Stories of P2P Streaming

streaming server

live event

Internet

peers
Challenges of P2P Streaming

Online users:

- high streaming rate
- shortest possible delay

Streaming server providers:

- simple to implement
- low protocol overhead
On achieving minimum delay
a centralized algorithm [Liu ACM Multimedia 07]

On achieving maximum streaming rate
a decentralized algorithm [Massoulie et al. INFOCOM 07]

On achieving near optimal steaming rate and delay
several decentralized algorithms [Bonald et al. SIGMETRICS 08]

However, none of them is actually implemented, why?
Practitioners: Pull-based Protocols

What has been implemented in practice, then? pull-based protocols

CoolStreaming [Zhang et al. INFOCOM 05]

However, no theoretical guarantees
Can We Bridge Theory and Practice?

One approach:

make the idealized algorithms practical

The other approach: more difficult

identify the major problem of pull-based protocols
but, much harder to model!

try to solve the major problem with new design
but, such a solution exists?

The more difficult something is, the more rewarding it is in the end.

— from Big Fish
The Goal of this Talk

Understanding pull-based protocols

identify the major problem
90% of the talk
solve the problem
10% of the talk
Understand Pull-based Protocols

How can we study pull-based protocols?

much harder to model than idealized algorithms

The idea: approximate the performance gap

step 1: establish the fundamental limits
step 2: identify key features of pull-based protocols
step 3: approximate the performance gap

between fundamental limits and actual performance
caused by key features of pull-based protocols
Lemma:

\[ R_{\text{max}} = U_p + \frac{U_s}{N} \]

\( U_s \) - server upload capacity
\( U_p \) - peer upload capacity
\( N \) - number of peers

Proof:

\[ NR \leq U_s + NU_p \]

[Kumar et al. INFOCOM 07]
Lemma: $D_{\text{min}} = \left\lfloor \log_2(N) \right\rfloor$

Proof: [Liu ACM Multimedia 07]

Assumptions:
- server: upload only one segment and then rest
- peers: one segment per time slot
Lemma: \( D_{\text{min}} = \lceil \log_2(N) \rceil \)

**Assumptions:**
- Server: upload only one segment and then rest
- Peers: one segment per time slot

**Proof:** [Liu ACM Multimedia 07]
Lemma: \( D_{\text{min}} = \lceil \log_2(N) \rceil \)

Assumptions:
- server: upload only one segment and then rest
- peers: one segment per time slot

Proof: [Liu ACM Multimedia 07]
Lemma: \( D_{\text{min}} = \lceil \log_2(N) \rceil \)

Assumptions:
- Server: upload only one segment and then rest
- Peers: one segment per time slot

Proof: [Liu ACM Multimedia 07]
Lemma: \( D_{\text{min}} \approx [\log_m(N)] \quad m = 1 + \mu_p \)

\( \mu_p \) expected value

Assumptions:

server: upload only one segment and then rest

peers: not one segment per time slot, but a random variable with expected value \( \mu_p \)

Proof: [INFOCOM 2009]

a branching process

Kesten and Stigum (1966)

\[ D_{\text{min}} \leq \log_m(N) - \log_m(W) + 1 \]

\[ D_{\text{min}} \geq \log_m(N) - \log_m(W) \]

\( W \) also a random variable
Fundamental Limits: Delay [3]

Theorem: [Liu ACM Multimedia 07] [INFOCOM 2009]

minimum delay of a single segment
= minimum delay of every segment

Assumption: server uploads a new segment every time slot
Theorem: [Liu ACM Multimedia 07] [INFOCOM 2009]

minimum delay of a single segment
= minimum delay of every segment

Assumption: server uploads a new segment every time slot
Fundamental Limits: Delay [3]

Theorem: [Liu ACM Multimedia 07] [INFOCOM 2009]

minimum delay of a single segment
= minimum delay of every segment

Assumption: server uploads a new segment every time slot
Fundamental Limits: Delay [3]

Theorem: [Liu ACM Multimedia 07] [INFOCOM 2009]

minimum delay of a single segment
= minimum delay of every segment

Assumption: server uploads a new segment every time slot
Bounds:

streaming rate: \[ R_{\text{max}} = U_p + \frac{U_s}{N} \]
delay: \[ D_{\text{min}} = \left\lceil \log_m \left( \frac{N}{u_s} \right) \right\rceil + 1, \quad m = 1 + u_p \]

Achievability:

there exists a graph labeling algorithm such that

\[ D = D_{\text{min}} \]
\[ R = U_p, \quad R \to R_{\text{max}} \text{ as } N \to \infty \]
Identify Key Features

In practice...

do not schedule every time slot, why?

overhead issues

no centralized scheduler
Identify Key Features

In pull-based protocols

CoolStreaming [Zhang et al. INFOCOM 05]
Identify Key Features

In pull-based protocols

periodically exchange segment availability

CoolStreaming [Zhang et al. INFOCOM 05]
Identify Key Features

In pull-based protocols

periodically exchange segment availability

request the missing segments independently

CoolStreaming [Zhang et al. INFOCOM 05]
Identify Key Features

In pull-based protocols

periodically exchange segment availability

request the missing segments independently

They are two key features!

CoolStreaming [Zhang et al. INFOCOM 05]
Feature 1: Periodic Exchanging Availability

periodic exchanging + centralized scheduling

Theorem: \( D_{\text{min}}(T) = T \left[ \log_{1 + u_p T}(N) \right] + T \)

Proof: generalized graph labeling algorithm [INFOCOM 2009]

\[
D_{\text{min}}(T) = T \left[ \log_{1 + u_p T}(N) \right] + T
\]
Feature 2: Distributed Scheduling

periodic exchanging + distributed scheduling

The idea:

design a distributed analogy of our generalized algorithm approximated by difference equations

\[ q_{k+1}(k + 1) = u_s T \]

\[ q_k(k + 1) = q_k(k) + u_p T m_k(k)(1 - \frac{q_k(k)}{N}) \]

\[ q_{k-i}(k + 1) = q_{k-i}(k) + u_p T m_{k-i}(k)(1 - \frac{q_{k-i}(k)}{N}) \]

\[ m_k(k) = q_k(k) \]

\[ m_{k-i}(k) = q_{k-i}(k) \prod_{j=0}^{i-1} \left(1 - \frac{q_{k-j}(k)}{N}\right) \]
Our Results: Delay

Periodic exchanging matters most!

- **limit**: fundamental limits
- **period**: periodic exchanging + centralized scheduling
- **simple**: periodic exchanging + distributed scheduling
Our Results: Streaming Rate

Periodic exchanging matters most!

limit: fundamental limits
period: periodic exchanging + centralized scheduling
simple: periodic exchanging + distributed scheduling
The Goal of this Talk

Understanding pull-based protocols

identify the major problem

90% of the talk

solve the problem

10% of the talk
One Possible Solution: Network Coding (NC)

- **upstream peers of p**
- **data**
- **on peer p**
- **downstream peers of p**

**PULL**

**CODING**
Goals Revisited

Understanding pull-based protocols

identify the major problem

periodic exchanging

solve the problem

one solution: network coding
Thank You!