Towards Hybrid Cloud-assisted Crowdsourced Live Streaming: Measurement and Analysis

Cong Zhang[*], Jiangchuan Liu[*], Haiyang Wang[+]
[*]Simon Fraser University, [+]University of Minesota Duluth

May, 2016

Outline

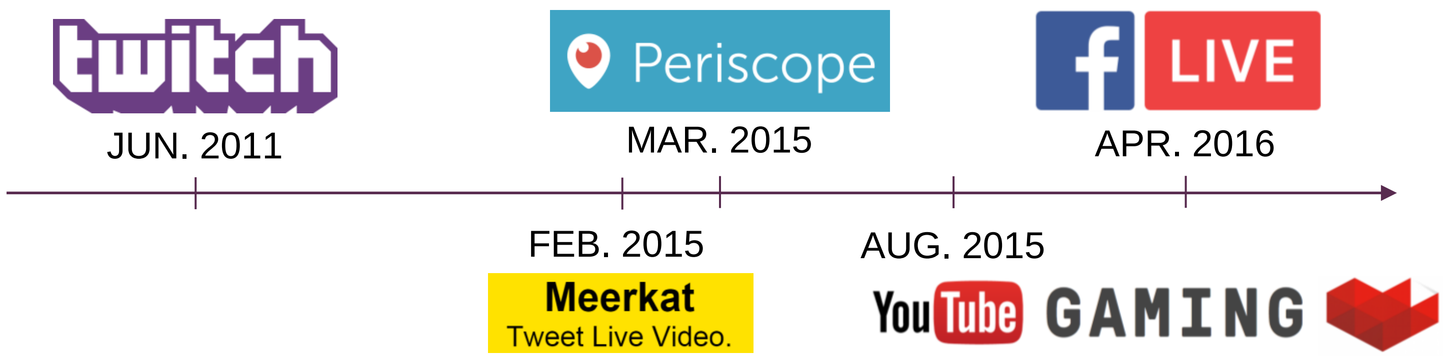
- Background
- Data description
- Measurement results
- HyCLS: Hybrid design and solution
- Trace-driven simulation and results
- Conclusion and further discussion

Outline

- **Background**
- Data description
- Measurement results
- HyCLS: Hybrid design and solution
- Trace-driven simulation and results
- Conclusion and further discussion

Background-1

Crowdsourced live streaming (CLS) has attracted a substantial amount of attentions from both industry and academia.



Due to the growth of e-sports games and the development of high-performance personal devices and networks, Twitch became the biggest crowdsourced live streaming platform.

Background-3

twitch

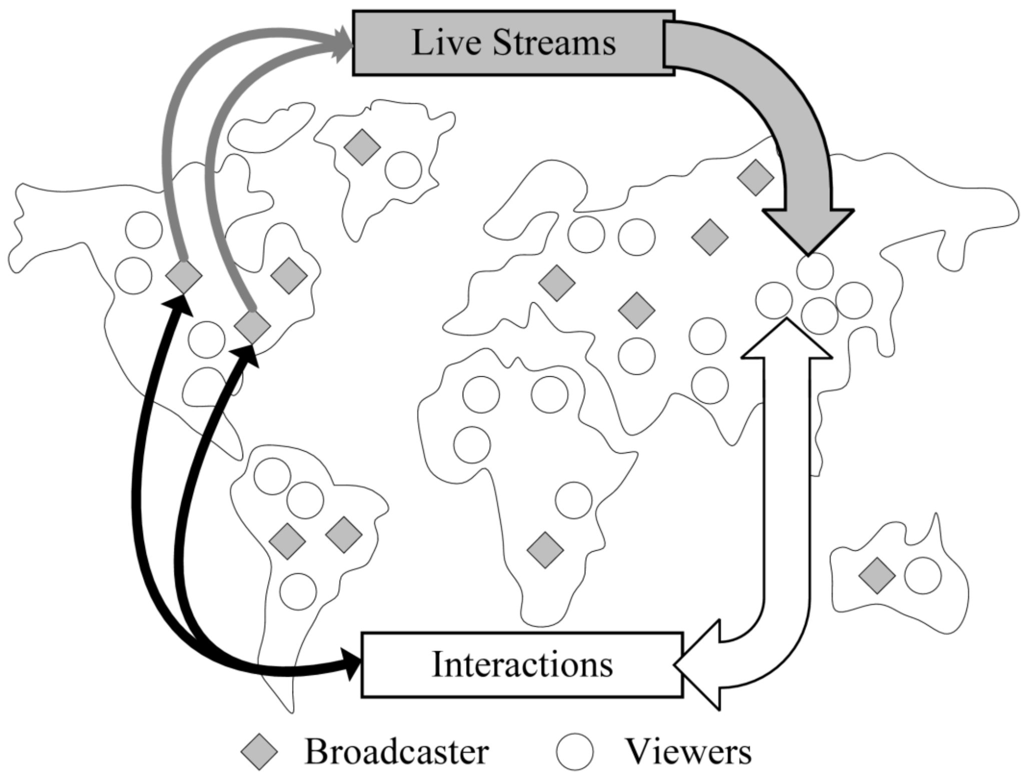
Number of broadcasters: **2.1 million**
Number of monthly streams: **11 million**
Number of monthly unique users: **100 million**
Amount of game content that has been streamed: **241 billion minutes**

Periscope

Number of total users: **10 million**
Number of daily active users: **2 million**
Number of broadcast to date: **200 million**
Amount of video content that is streamed daily: **350,000 hours of video**

Background-2

The generic framework of CLS.



Outline

- Background
- **Data description**
- Measurement results
- HyCLS: Hybrid design and solution
- Trace-driven simulation and results
- Conclusion and further discussion

Data Description

- Broadcaster datasets
  - Total number of views
  - Playback bitrate, resolution, partner status
  - About **1.5 million** broadcasters
- Stream datasets
  - The number of viewers per five minutes
  - Start time, duration, game name
  - About **9 million** streams

Outline

- Background
- Data description
- **Measurement results**
- HyCLS: Hybrid design and solution
- Trace-driven simulation and results
- Conclusion and further discussion

Measurement Result-1

- CLS highlights the event-related live streams with different broadcasters.
- In each CLS event, streaming contents have an event-based correlation, but show broadcaster-based differences.

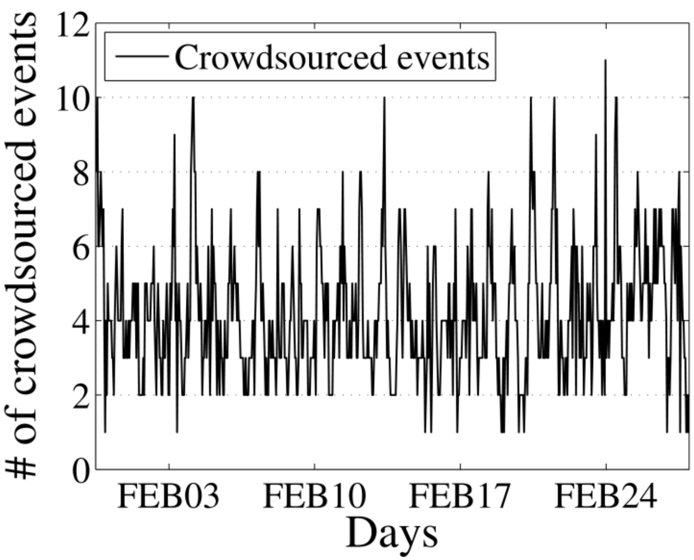


Fig. 2a Daily pattern

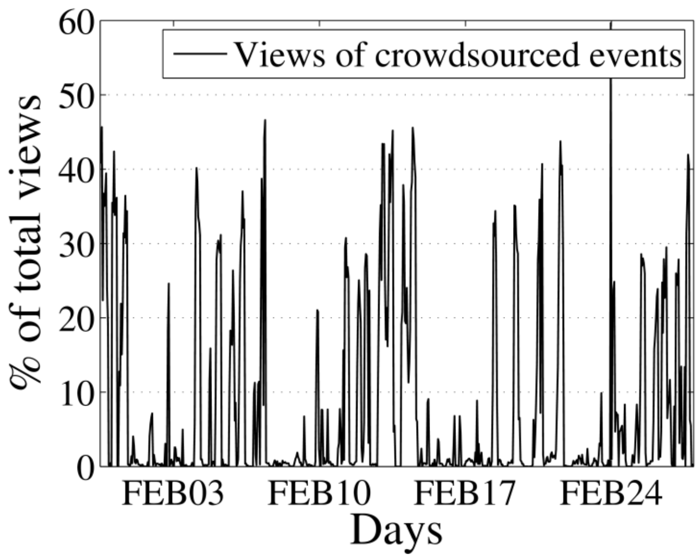


Fig. 2b Effects of crowdsourced live events

Measurement Result-2

- We also explore the popularity of broadcasters.
- We plot the highest number of concurrent views against the rank of the broadcasters (in terms of the popularity) in log-log scale.

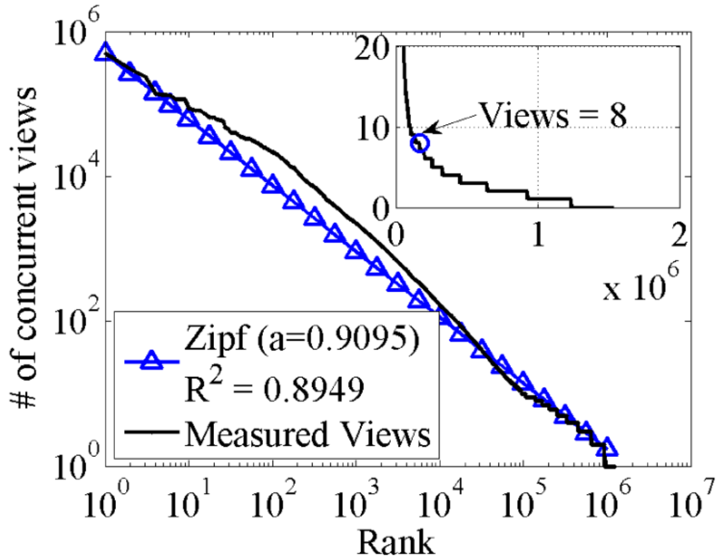


Fig. 3 Broadcasters rank ordered by popularity

Measurement Result-3

- The distribution of live duration

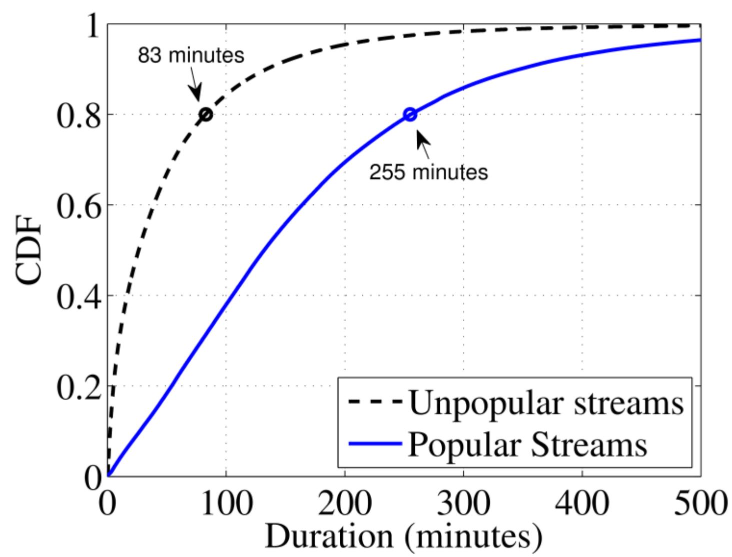


Fig. 4 The distribution of live duration

The total duration of all unpopular streams in one month is nearly 830 years, while the total duration of popular streams is only 310 years.

Measurement Result-4

- The daily activity of two broadcasters

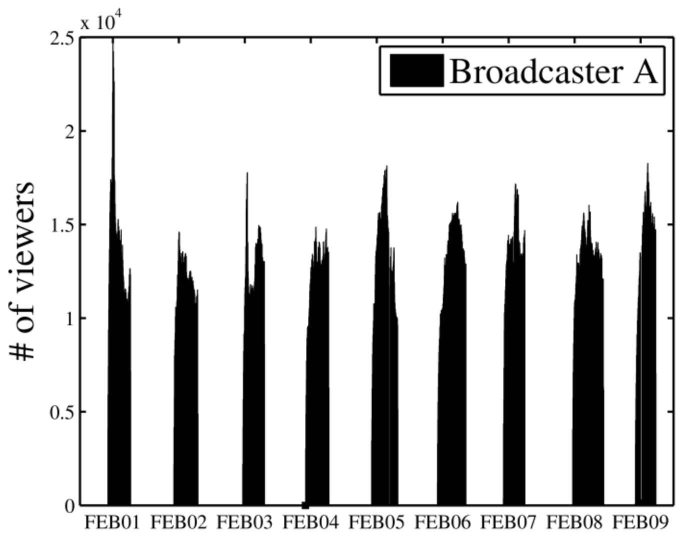


Fig. 5a Popular broadcaster sample

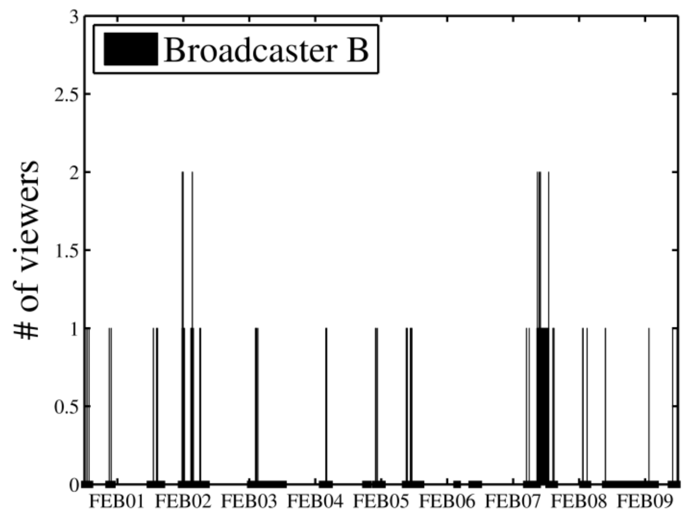


Fig. 5b Unpopular broadcaster sample

A: regular live schedule, stable live duration, a large number of viewers.
B: dynamic schedule and duration, a few number of viewers

Measurement Result-5

- Broadcaster arrivals per five minutes.

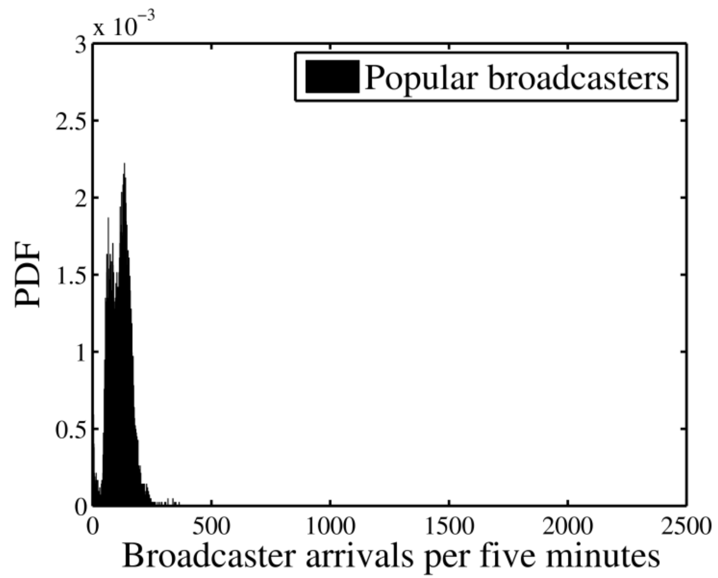


Fig. 6a Popular broadcasters

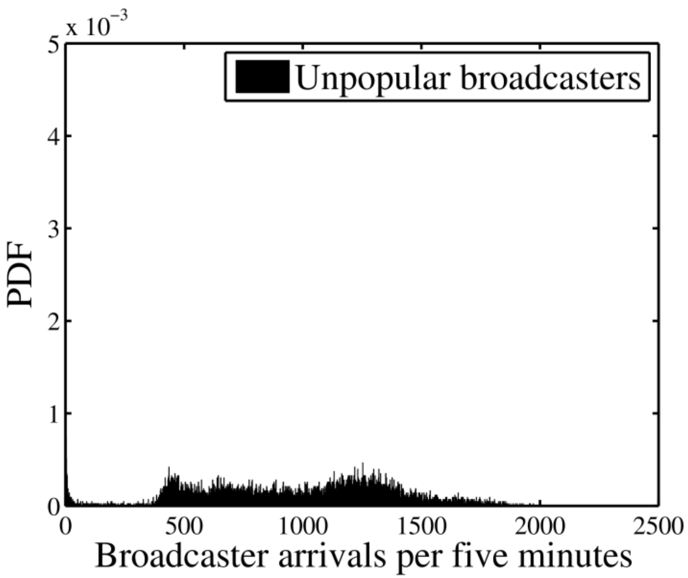


Fig. 6b Unpopular broadcasters

CLS features

CLS feature:

- 1. Live sources
  - Controlled by broadcasters vs. Managed by service providers
- 2. Service cost
  - Storage/bandwidth/.computation continually vs. storage

How about the resource consumption of hosting these unpopular broadcasters in Twitch?

Measurement Result-6

- The effectiveness of resource consumption.

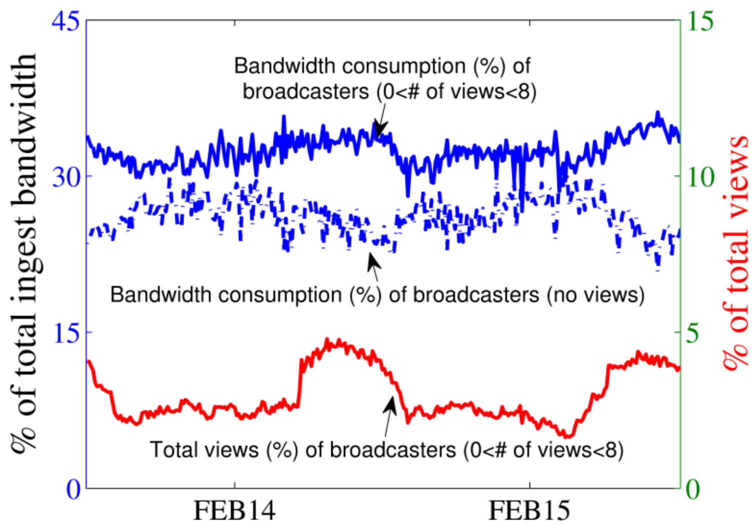


Fig. 7a Bandwidth consumption

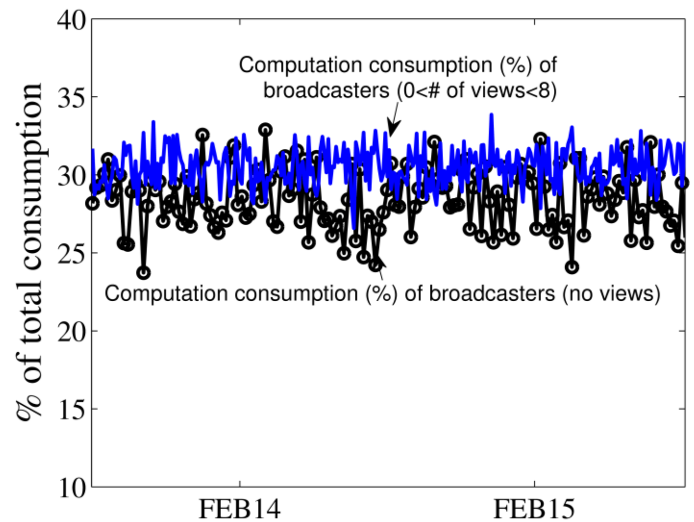


Fig. 7b Computation consumption

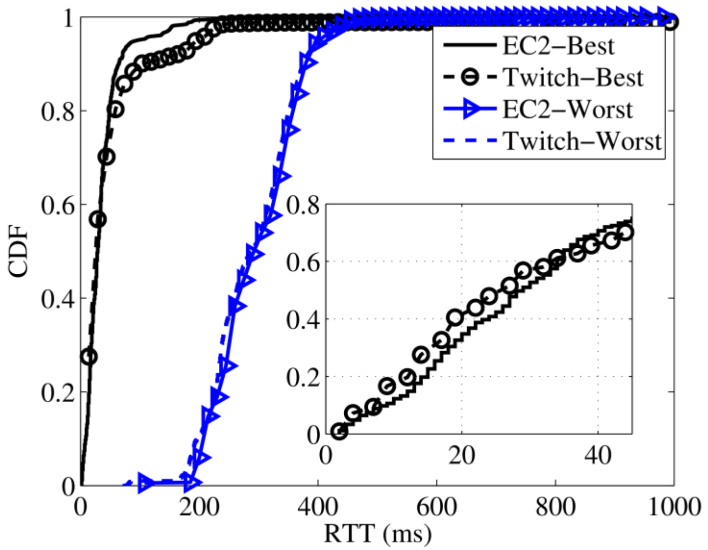
R. Aparicio-Pardo, K. Pires, A. Blanc, and G. Simon. Transcoding live adaptive video streams at a massive scale in the cloud. In *ACM MMSys, 2015.*

Analysis

- Crowdsourced live events
- Unpopular broadcasters
  - Dynamic schedule
  - Unstable live duration
  - Frequent arrival
- Dedicated resource consumption
  - Bandwidth
  - Computation
- Can we use public cloud to assist existing private datacenter ?

Measurement Result-7

- RTT comparison between public cloud and private data center

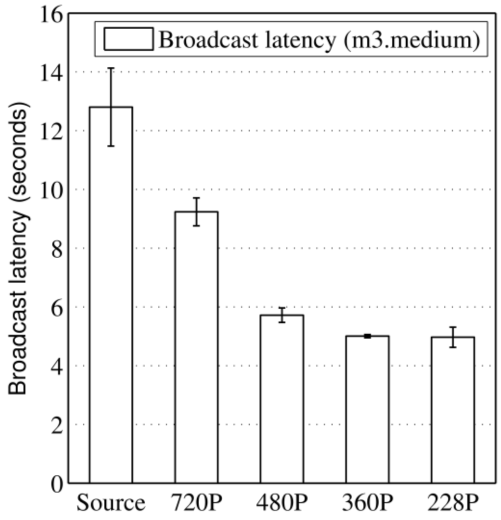


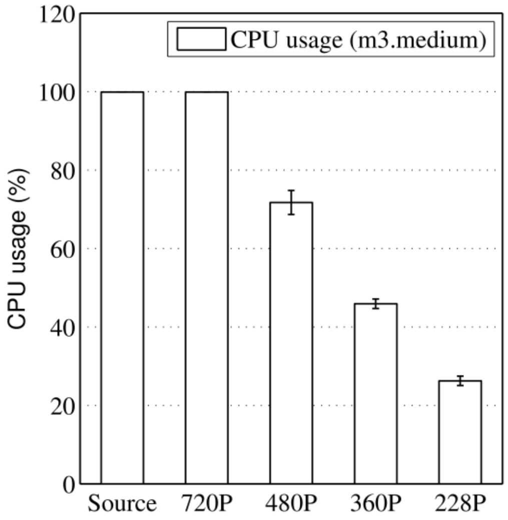
EC2 instances do not increase RTT significantly even in the degradation of networks.

We can use EC2 instance to ingest the live streams of broadcasters without extra latency.

Measurement Result-8

- Performance comparison between different instance types (m3.medium vs. m3.large)

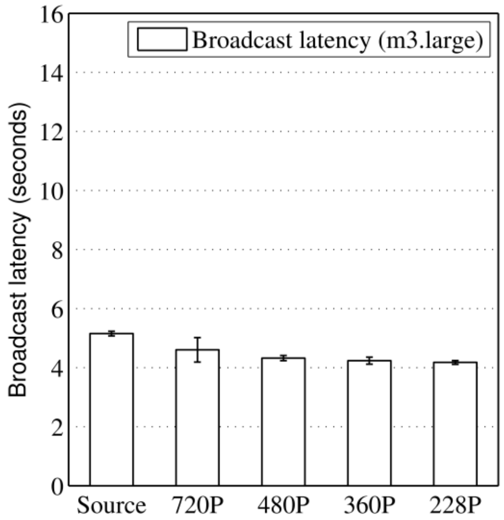


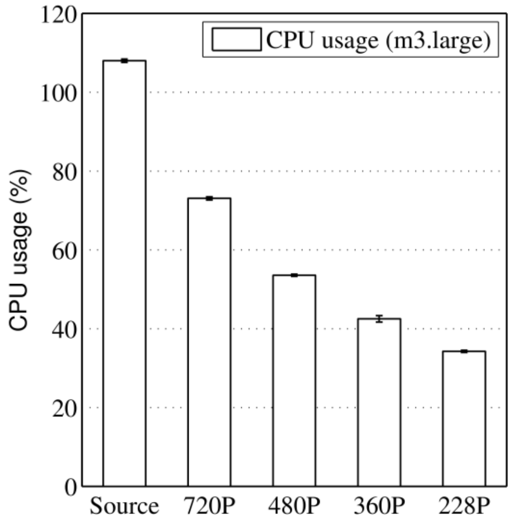


Source(i.e., 1080P, 3200Kbps)
720P (1500Kbps),
480P (800Kbps)
360P (500Kbps)
228P (200Kbps)

Measurement Result-9

- Performance of different types of instance (m3.large).



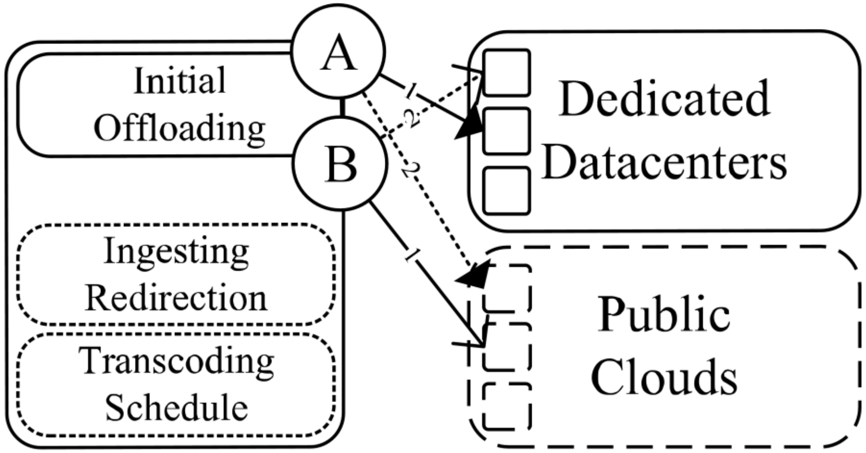


Outline

- Background
- Data description
- Measurement results
- **HyCLS: Hybrid design and solution**
- Trace-driven simulation and results
- Conclusion and further discussion

HyCLS Design

- Our goal is to assign broadcasting workloads cost-effectively in hybrid-design.



HyCLS-Initial Offloading

- Stable Index reflects the similarity of b’s resource consumption in recent n days.
- Update threshold periodically to determine the initial offloading.

$$SI^{(b)} = \begin{cases} \frac{1}{n} \sum_{i=2}^{n} \frac{\sum_{j=1}^{m} d_{i,j}^{(b)} \cdot d_{i-1,j}^{(b)}}{\sum_{j=1}^{m} d_{i-1,j}^{(b)}} & \text{if } \sum_{j=1}^{m} d_{i-1,j}^{(b)} \neq 0 \\ 0 & \text{otherwise} \end{cases}$$

Ingesting Redirection & Transcoding Schedule

Utility function: $U^{(t)}(b,r) = \sum_{v \in V} G^{(t)}(b,r,v) \cdot N^{(t)}_{(b,v)}$

The number of viewers

$$G^{(t)}(b,r,v) = \alpha + ln(1 - \beta L^{(t)}_{(b,r,v)})$$

Gain

Transcoding Latency

Broadcast Latency $L^{(t)}_{(b,r,v)} = l^{(t)}_{(b,r)} + l^{(t)}_{(q_b,q_v)} + l_{(r,v)}$

Ingest Latency

Distribute Latency

Ingesting Redirection & Transcoding Schedule

$$\underset{e \in E^{(t)}}{\text{Maximize}}\ F(A^{(t)}) = \min_{\substack{b \in e \\ r \in R}} \{U^{(t)}(b, r)\}$$

Resource Availability Constraints:

$$\forall r \in R, W_r^{(t)} \leq \mathbb{W}_r$$

$$\forall r \in R, C_r^{(t)} \leq \mathbb{C}_r$$

Budget Constraints:

$$\sum_{r \in R} \frac{W_r^{(t)}}{\mathcal{W}} \cdot Cost_w(r) \cdot I(r) \leq K_w$$

$$\sum_{r \in R} \frac{C_r^{(t)}}{\mathcal{C}} \cdot Cost_c(r) \cdot I(r) \leq K_c$$

Ingesting Redirection & Transcoding Schedule

$$\underset{e \in E^{(t)}}{\text{Maximize}}\ F(A^{(t)}) = \min_{\substack{b \in e \\ r \in R}} \{U^{(t)}(b, r)\}$$

Resource Availability Constraints:

$$\forall r \in R, C_r^{(t)} \leq \mathbb{C}_r$$

Budget Constraints:

$$\sum_{r \in R} \frac{C_r^{(t)}}{\mathcal{C}} \cdot Cost_c(r) \cdot I(r) \leq K_c$$

H. Wang, R. Shea, X. Ma, F. Wang, and J. Liu. On design and performance of cloud-based distributed interactive applications. In *IEEE ICNP, 2014.*

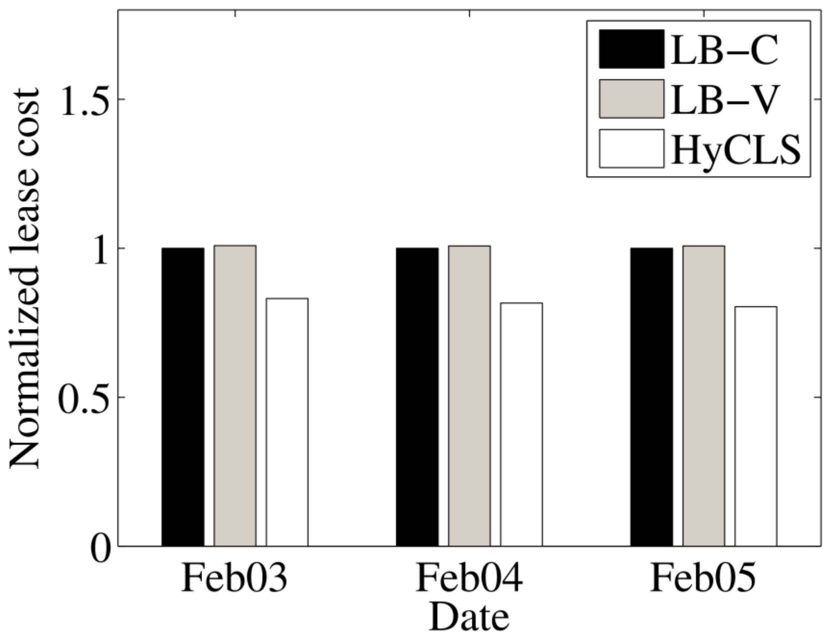
Outline

- Background
- Data description
- Measurement results
- HyCLS: Hybrid design and solution
- **Trace-driven simulation and results**
- Conclusion and further discussion

Simulation setup

- Partner status in Twitch
- Homogenous public instances (m3.large)
- $\alpha = 1$ and $\beta = 0.011$: make the gain $G(t)(\cdot) \in [0, 1]$ with current Twitch broadcast latency interval.
- $n = 2$: Stable index is calculated by using the data-trace during latest two days.

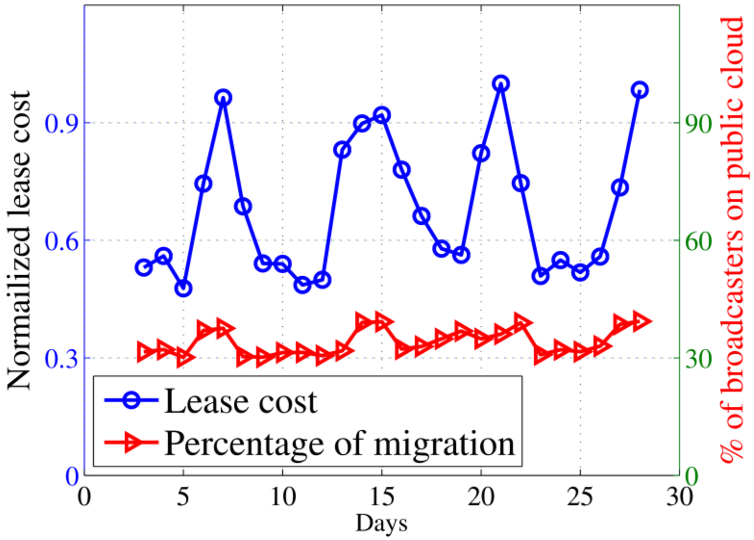
Simulation Results-1



- Views-based (LB-V): only considers the current number of views in different live streams;
- Computation-based (LB-C): migrates workload based on the consumption of computation resources.

. Our HyCLS-based approach has the lowest cost, decreasing 16.9%-19.5% of LB-C approach and 17.8%-20.4% of LB-V approach.

Simulation Results-2



The daily lease cost performs the weekly pattern and provide elastic workload provisioning cost-effectively. Moreover, more than 30% of broadcasters are migrated to the public cloud in every day.

Outline

- Background
- Data description
- Measurement results
- HyCLS: Hybrid design and solution
- Trace-driven simulation and results
- **Conclusion and further discussion**

Conclusion and Further Discussion

- The characteristics of broadcasters in Twitch
- The challenges of bandwidth and computation comsuption
- Hybrid-cloud design and solution

- Re-design initial offloading strategy
- Amazon EC2 and PlanetLab-based practical deployment

Thank You!

Q&A