Optimal Strategies for Live Video Streaming in the Low-latency Regime

Liyang Sun, Tongyu Zong, Yong Liu and Yao Wang, New York University

Haihong Zhu, Futurewei Technologies

Video-over-Ideal-5G

Application
Transport
Network
Data Link
Physical

24K, 360 Degree, Volumetric

AR/VR/MR

Live/Interactive

Higher Throughput: Gbps

Lower Latency, 1ms

Video-over-5G: real challenges

Application Layer	User QoE Optimization with Realistic Network Assumptions
Transport	
Network	
Data Link	
Physical	

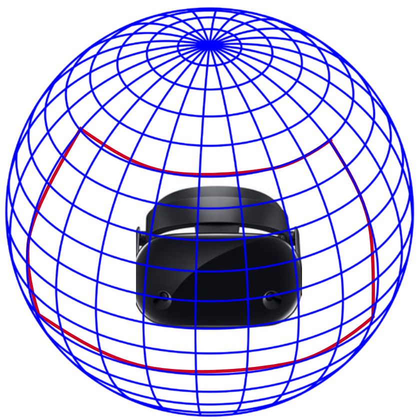
- ❖ Users sensitive to video quality and temporal variation
- ❖ Video freezes/skips/black-screen detrimental to user QoE
- ❖ Long end-to-end video delay kills interactivity
- ❖ Users want mobile/wireless video

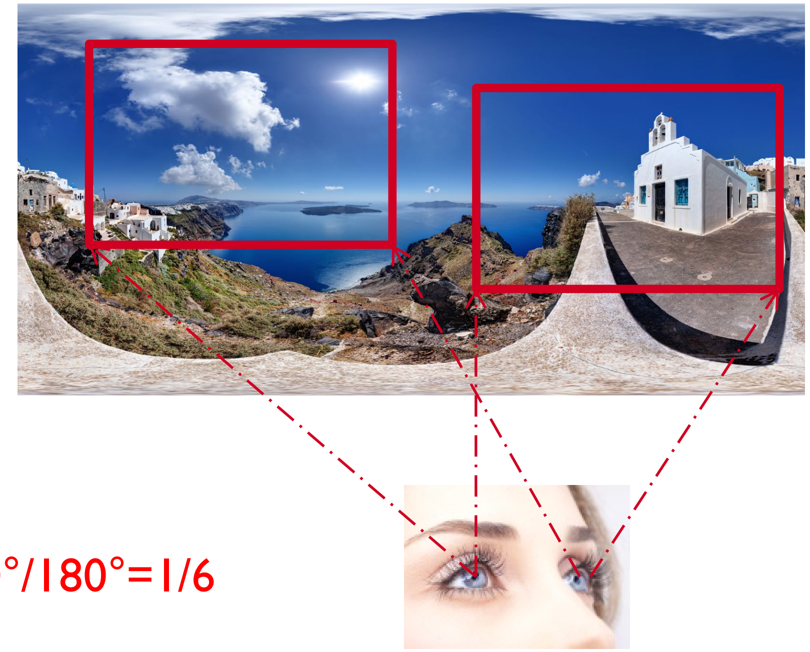
??? Consistently High-throughput/Low-delay from Lower Layers ???

360-degree Video Streaming Projects (joint with Yao Wang)

1. Two-tier on-demand 360° video streaming

- Field-of-View (FoV) streaming to reduce b.w. requirement (1/6)
- two-tier segment coding/streaming to be robust against b.w. and FoV dynamics

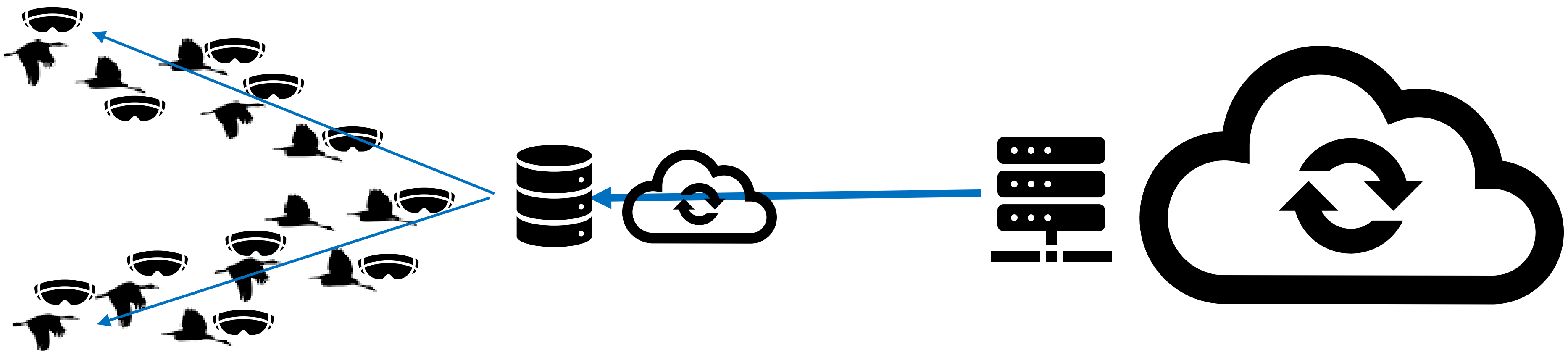




Fraction of area within FoV: 120°/360° x 90°/180°=1/6

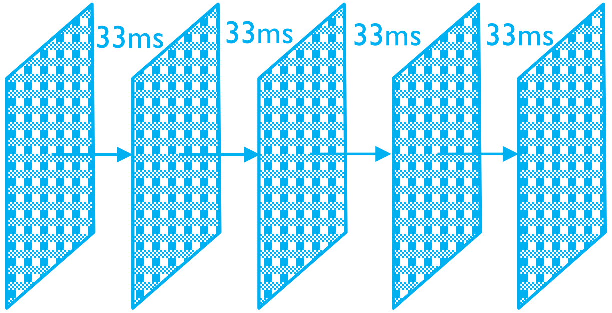
360-degree Video Streaming Projects (joint with Yao Wang)

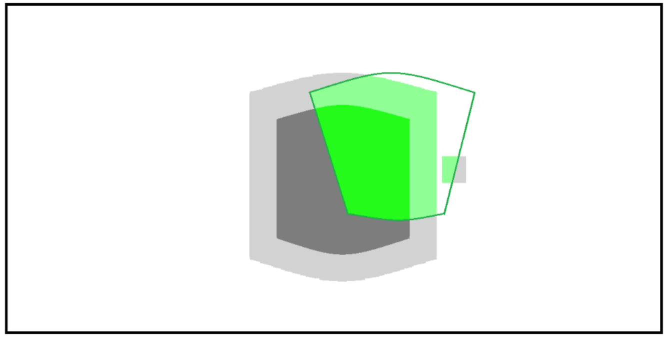
1. Two-tier on-demand 360° video streaming
   - Field-of-View (FoV) streaming to reduce b.w. requirement (1/6)
   - two-tier segment coding/streaming to be robust against b.w. and FoV dynamics
2. Flocking-based live 360° streaming from edge cloud
   - users watching same live event form a "flock"
   - users with shorter video lag lead flock: populating cache, generating realtime "saliency" map



360-degree Video Streaming Projects (joint with Yao Wang)

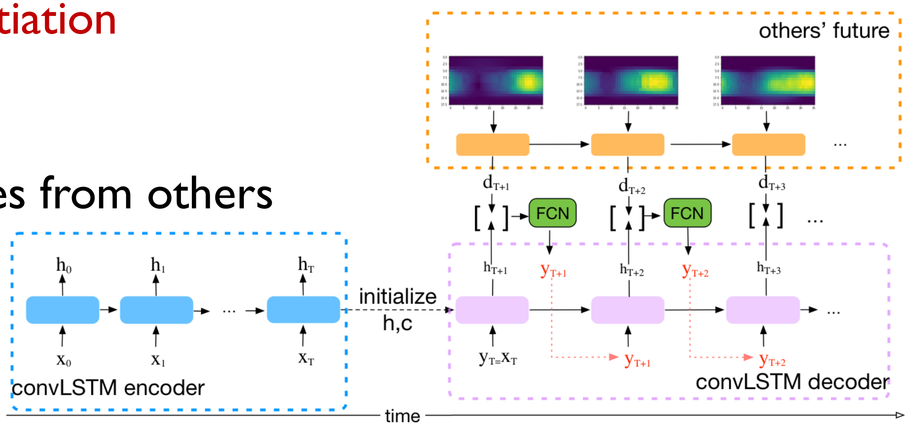
1. Two-tier on-demand 360° video streaming
   - Field-of-View (FoV) streaming to reduce b.w. requirement (1/6)
   - two-tier segment coding/streaming to be robust against b.w. and FoV dynamics
2. Flocking-based live 360° streaming from edge cloud
   - users watching same live event form a "flock"
   - users with shorter video lag lead flock: populating cache, generating realtime "saliency" map
3. Realtime coding and delivery for interactive streaming
   - frame-level coding and delivery to ensure tens milli-second latency
   - tile-based video rate allocation for FoV quality differentiation



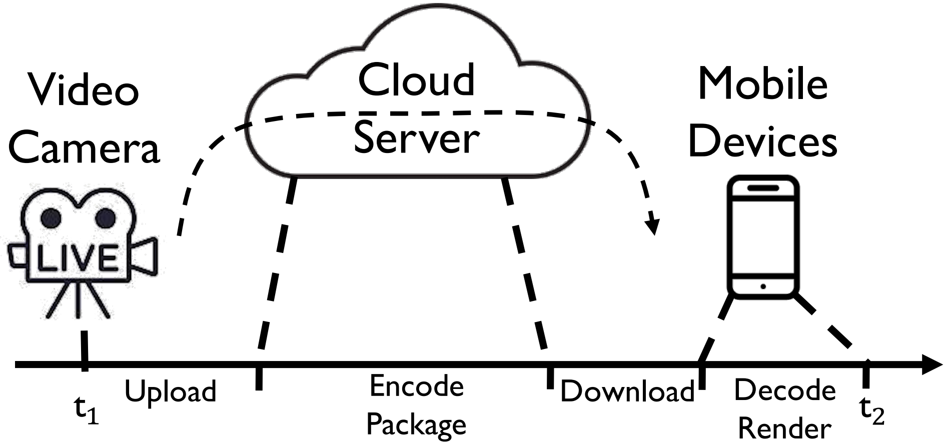


360-degree Video Streaming Projects (joint with Yao Wang)

1. **Two-tier on-demand 360° video streaming**
   - Field-of-View (FoV) streaming to reduce b.w. requirement (1/6)
   - two-tier segment coding/streaming to be robust against b.w. and FoV dynamics
2. **Flocking-based live 360° streaming from edge cloud**
   - users watching same live event form a "flock"
   - users with shorter video lag lead flock: populating cache, generating realtime "saliency" map
3. **Realtime coding and delivery for interactive streaming**
   - frame-level coding and delivery to ensure tens milli-second latency
   - tile-based video rate allocation for FoV quality differentiation
4. **Deep-learning based user FoV prediction**
   - target user past FoV trajectory, and "future" trajectories from others
   - video content saliency map



Low Latency Live Streaming



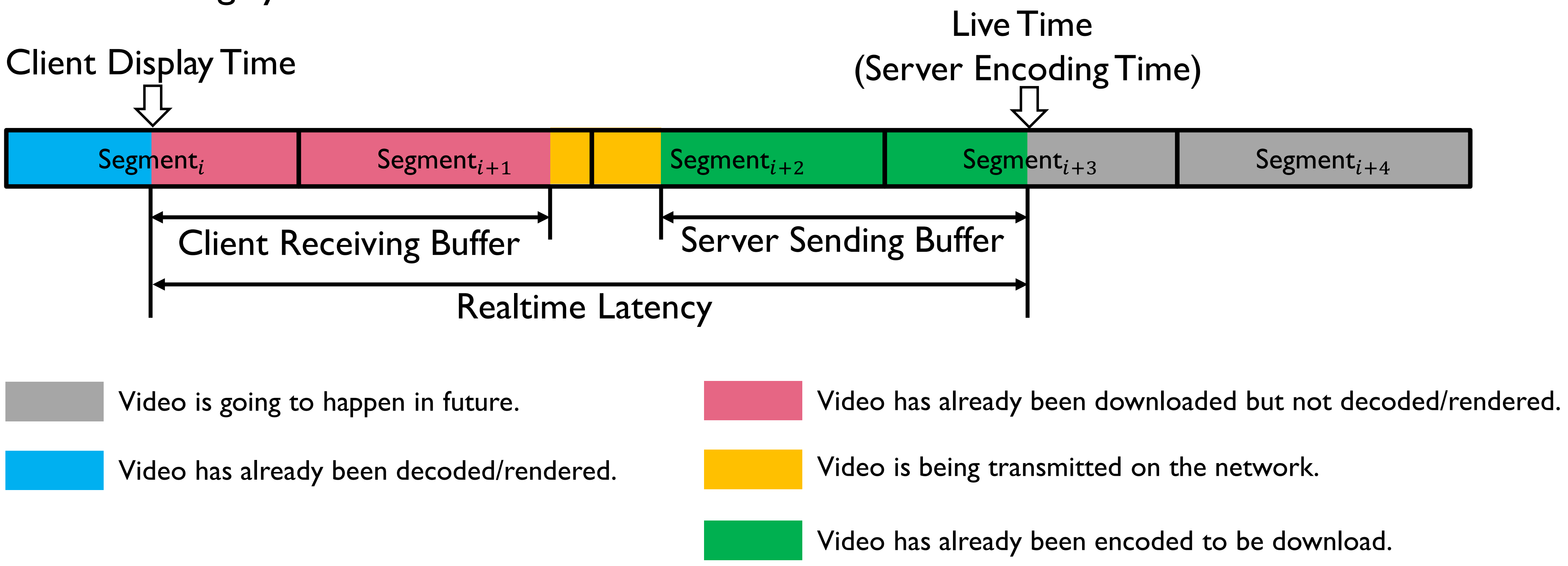
Application	Latency (s)
YouTube Live*	7-11
Facebook Live*	~15
Twitch*	~15
FOX, abc**	~7

*: Online Live Streaming
**: TV broadcasting

- Sports, online gaming broadcast, social live UGC,
- Online live streaming still lags behind TV.
- User live/interactivity experience is ruined by long latency!
- Can we simply shorten latency in live streaming system? No!

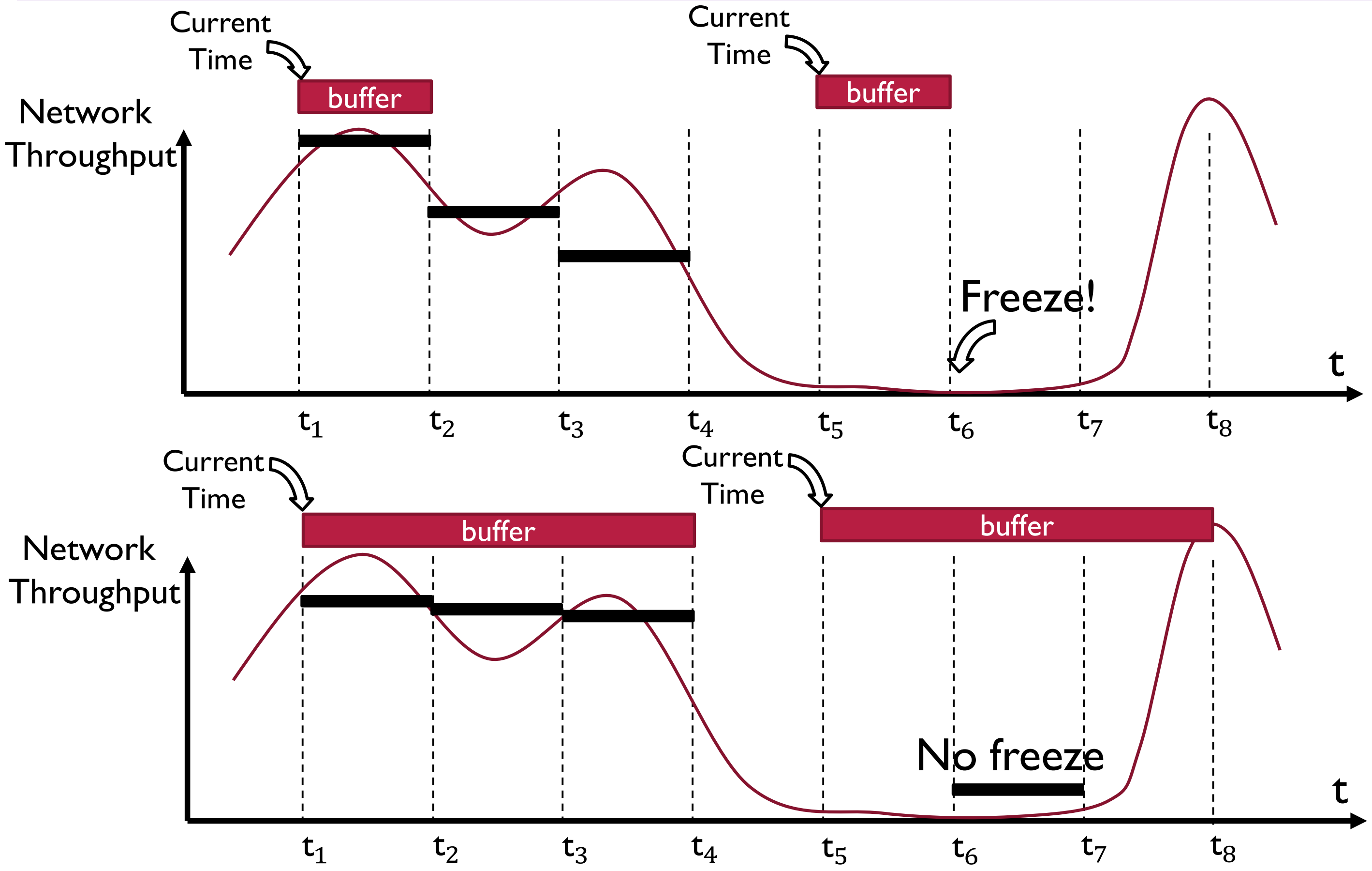
Low Latency and Buffer Length

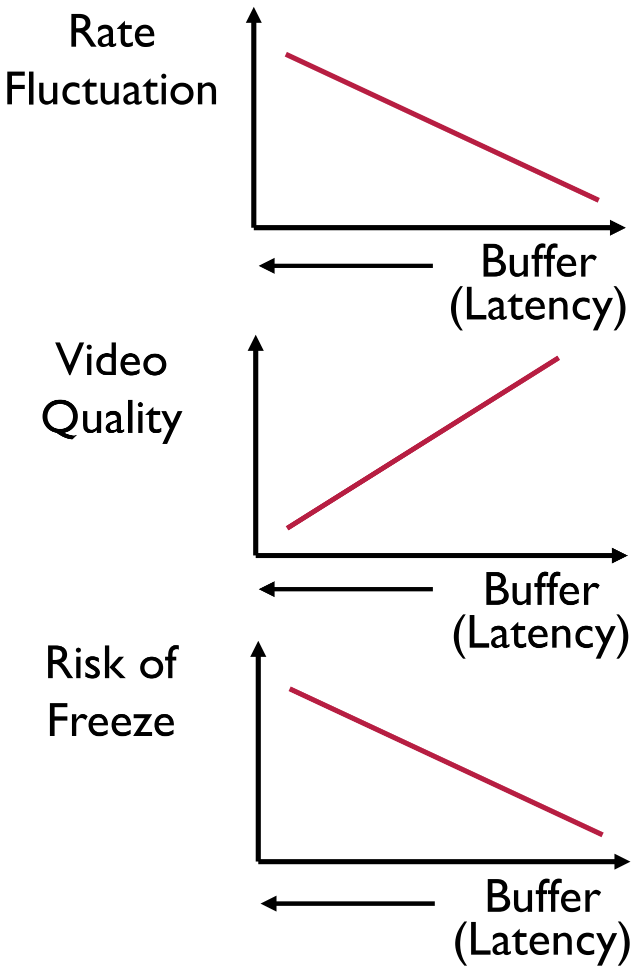
- Live streaming system state at time T



- Realtime latency is the upper bound of client buffer length.

Influence of Buffer Length (Latency)





- Shortening latency negatively impact all the other QoE metrics.
- Goal: Trade-off between latency and other metrics to maximize QoE.

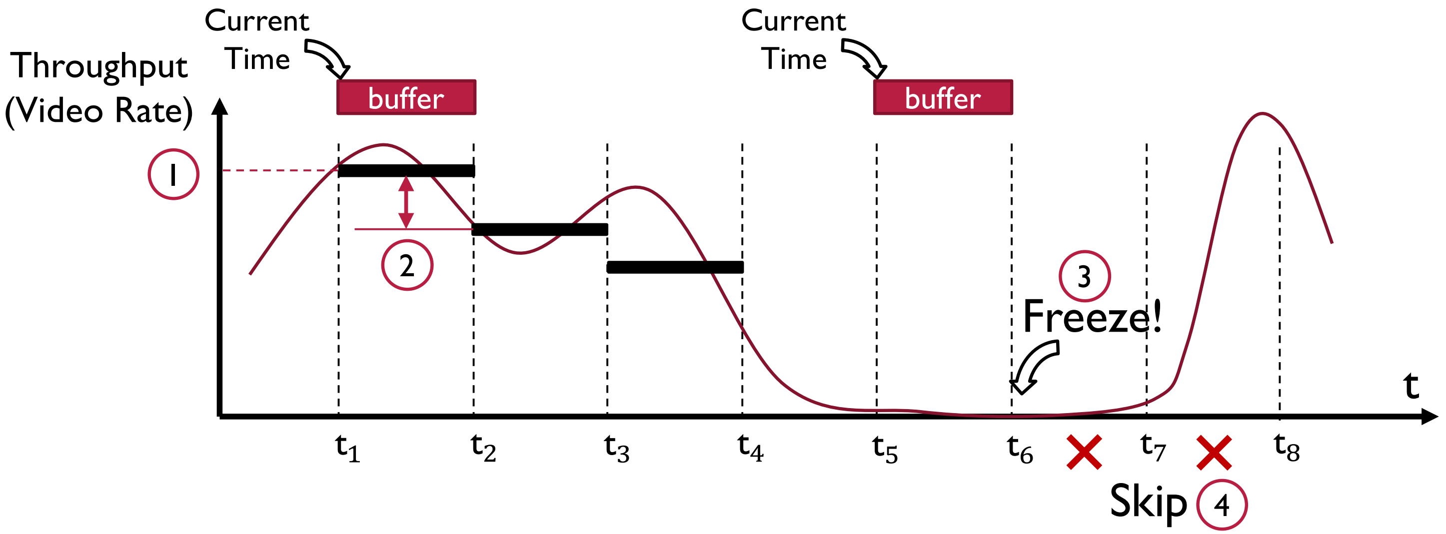
Live Streaming QoE

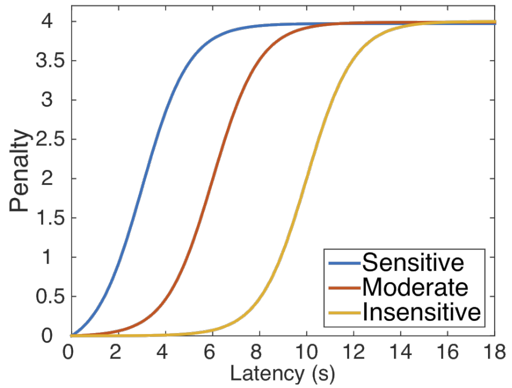
- QoE Metrics:

$$QoE = \underbrace{a_1 Q(r_i)}_{1} - \underbrace{a_2 |Q(r_i) - Q(r_{i-1})|}_{2} - \underbrace{a_3 x_i}_{3} - \underbrace{a_4 n_i}_{4} - \underbrace{a_5 \boldsymbol{g}(l_i)}_{5}$$

1 Video Rate$^+$ 2 Rate Fluctuation$^-$ 3 Freeze$^-$ 4 Skip$^-$ 5 Latency$^-$

$$\frac{1}{1+e^{\phi - l_i}} - \frac{1}{1+e^{\phi}}$$

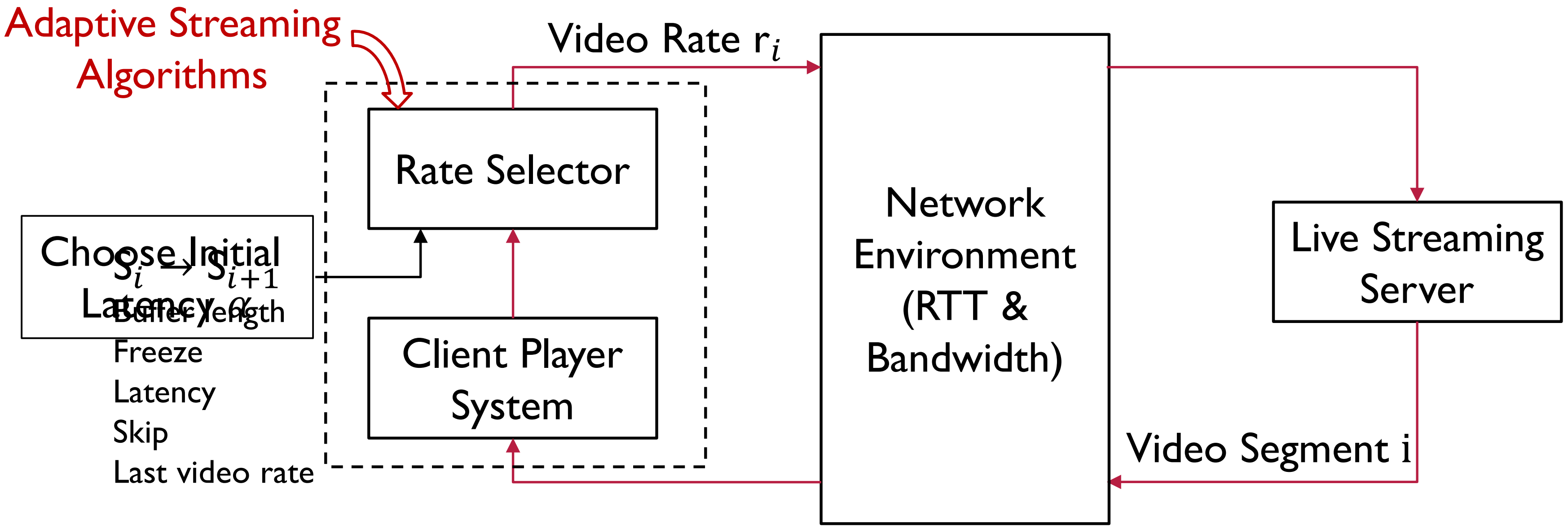




Model of Live Streaming System

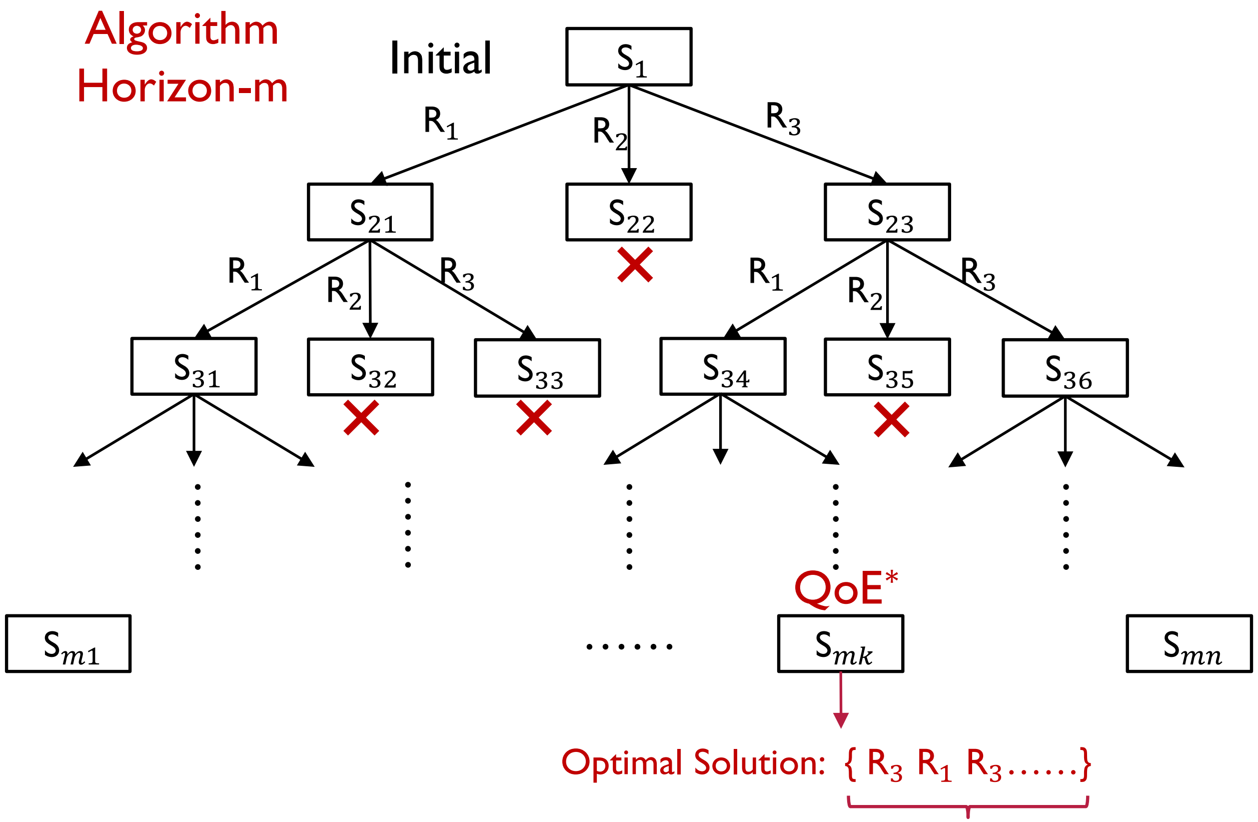
- System Evolution

Choose Initial Latency → Choose Rate → Download → Update System State → Choose Rate …



Optimal Streaming with Network Oracle

- Network condition for future m steps is available.



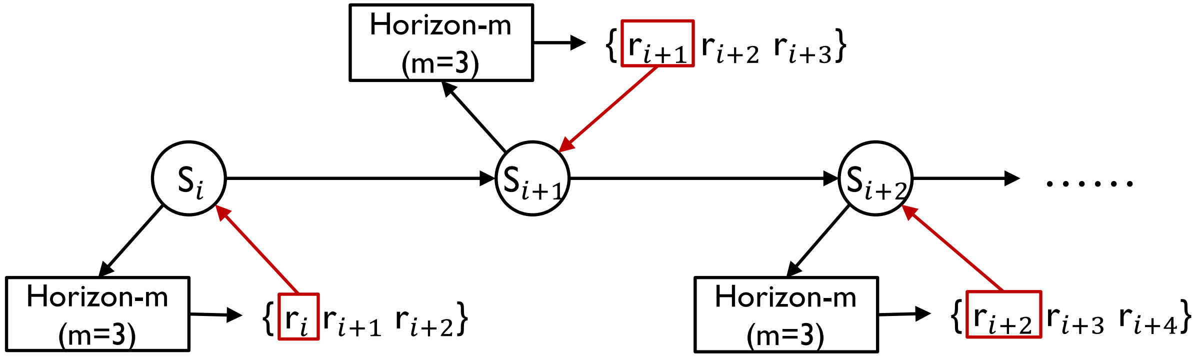
Algorithm 1 Optimal Streaming for Horizon-m

Input: $\mathcal{S}_1$: the initial state; m: look-ahead horizon; $\{w_i, rtt_i, i \in [1,m]\}$: future available bandwidth and rtt; $\mathcal{R}$: available rates;
Output: $\{r_i^*, i \in [1,m]\}$: optimal rate sequence.
Initialization: The possible states at stage 1: $\Omega_1 = \{\mathcal{S}_1\}$.

```
Branch-and-Bound State Expansion
for each segment i ∈ [1, m] do
    Ω_i = ∅
    for each state S in Ω_{i−1} do
        for each R_j ∈ R do
            S'_i = f(S, R_j, {w_i, rtt_i})
            if S'_i could be part of the overall optimal solution then
                Ω_i ← Ω_i ∪ S'_i
            end if
        end for
    end for
end for
Find Optimal Transition S_1 --r*_1--> S*_2 ∈ Ω_2 ··· --r*_{m+1}--> S*_{m+1} ∈ Ω_{m+1} to maximize accumulated QoE Σ_{i=1}^{m} QoE(S_i, r_i) through DP.
return r*_{[1,···,m]}
```

- Large m causes state explosion.

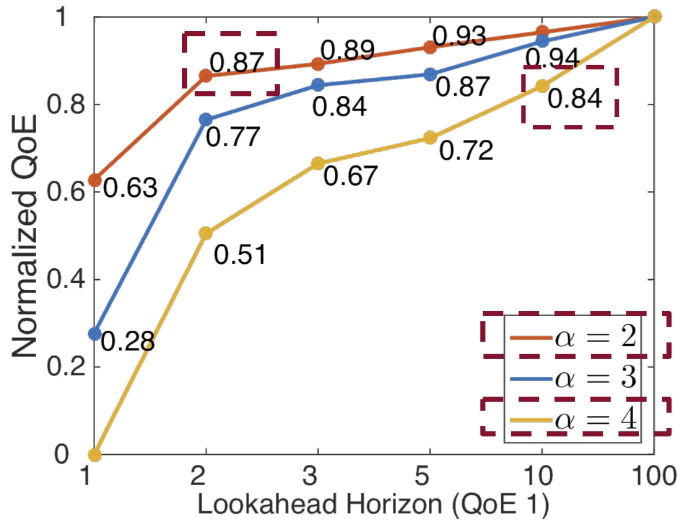
Sliding Window with Horizon-(Small m)



Algorithm 2 Sliding Horizon-m Streaming

Input: $\mathcal{S}_1$: initial state; α and β: startup parameters; m: look-ahead horizon; N: live streaming duration; $\{w_i, rtt_i, i \in [1, N]\}$: available bandwidth and rtt; $\mathcal{R}$: available rates.
Output: $\{r_i, i \in [1, N]\}$: rate sequence for all segments

Download the first β segments using predefined rate selection strategy $r_{[1,\cdots,\beta]}$, obtain $\mathcal{S}_{\beta+1}$
for each segment $i \in [\beta+1, N]$ **do**
  $rr_i^{(m)}$ =Horizon-m($\mathcal{S}_i, m, \{w_{[i,i+m-1]}, rtt_{[i,i+m-1]}\}, \mathcal{R}$)
  $r_i = rr_i^{(m)}[1]$
  $\mathcal{S}_{i+1} = \boldsymbol{f}(\mathcal{S}_i, r_i, \{w_i, rtt_i\})$
end for
return $r_{[1,\cdots,N]}$

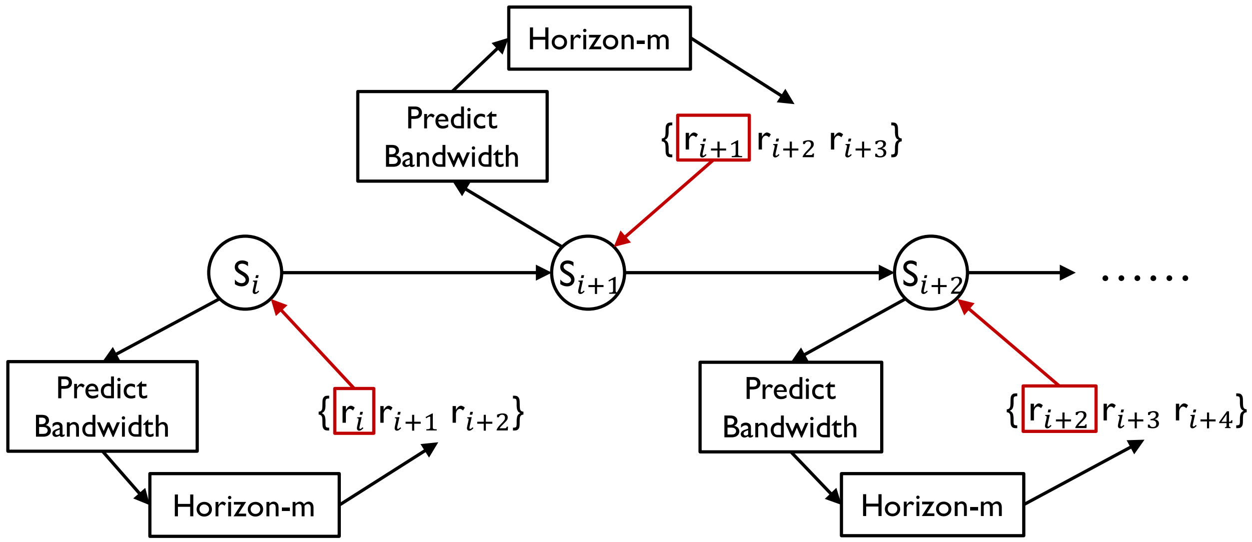


- When latency α=2, small lookahead horizon (m=2) is needed to get high QoE.
- If α=4, similar normalized QoE is achieved when m=10.

If latency is short, future information of short lookahead horizon is needed to achieve close-to-optimal QoE.

Model Predictive Control (MPC) for Live Streaming

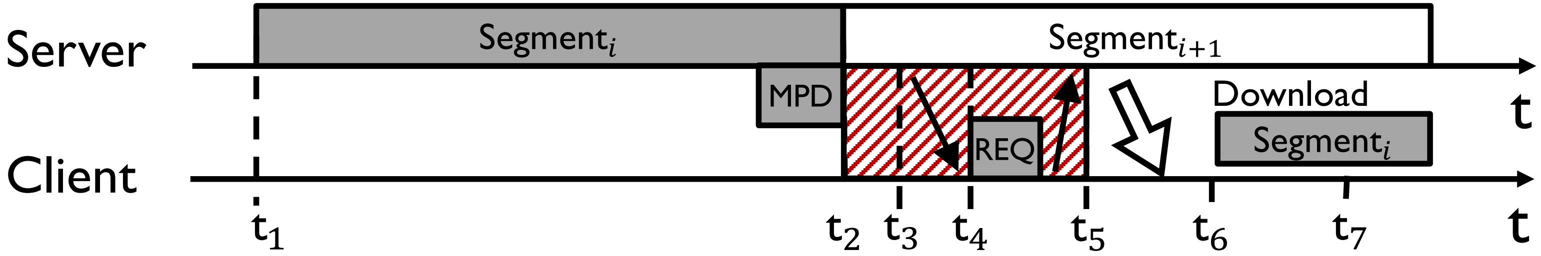
- Future network information is NOT available.
- Bandwidth predictions
  - Harmonic Mean, Hidden Markov Model (HMM), Recursive Least Squared (RLS) and LSTM.



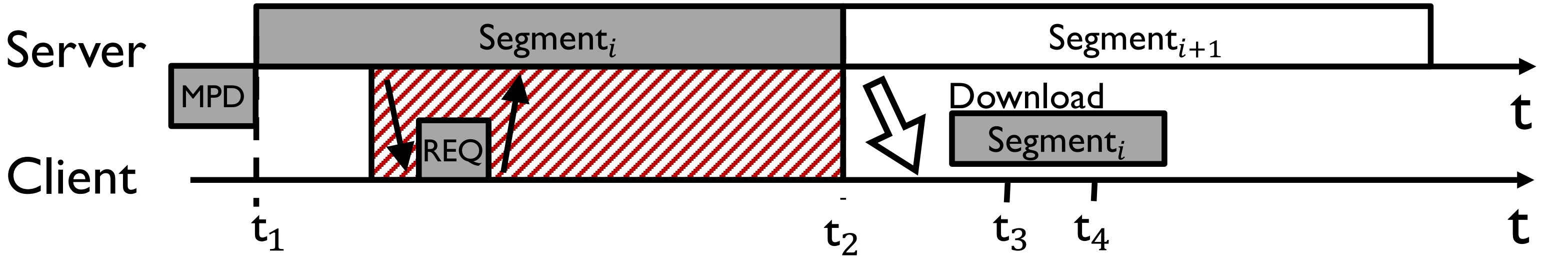
MPC based Practical Live Streaming Algorithm

Segment & Chunk based Streaming

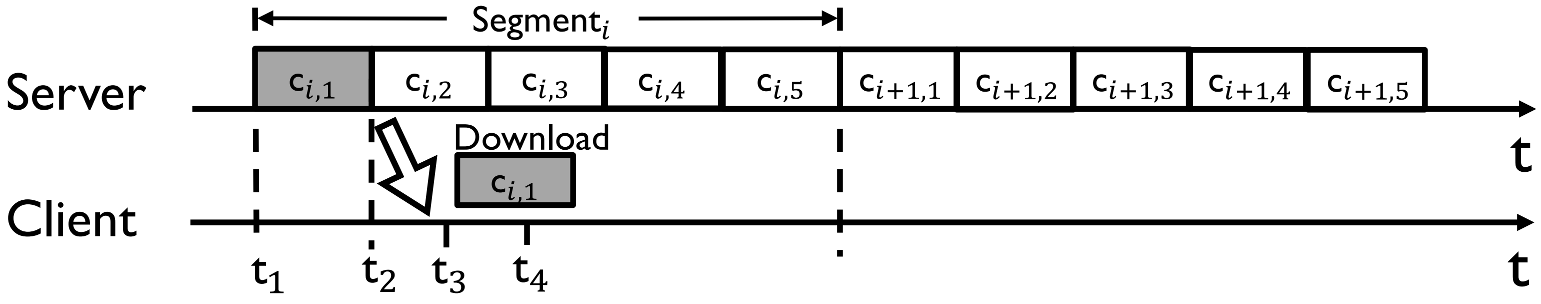
- Segment Mode



- Server-wait Mode

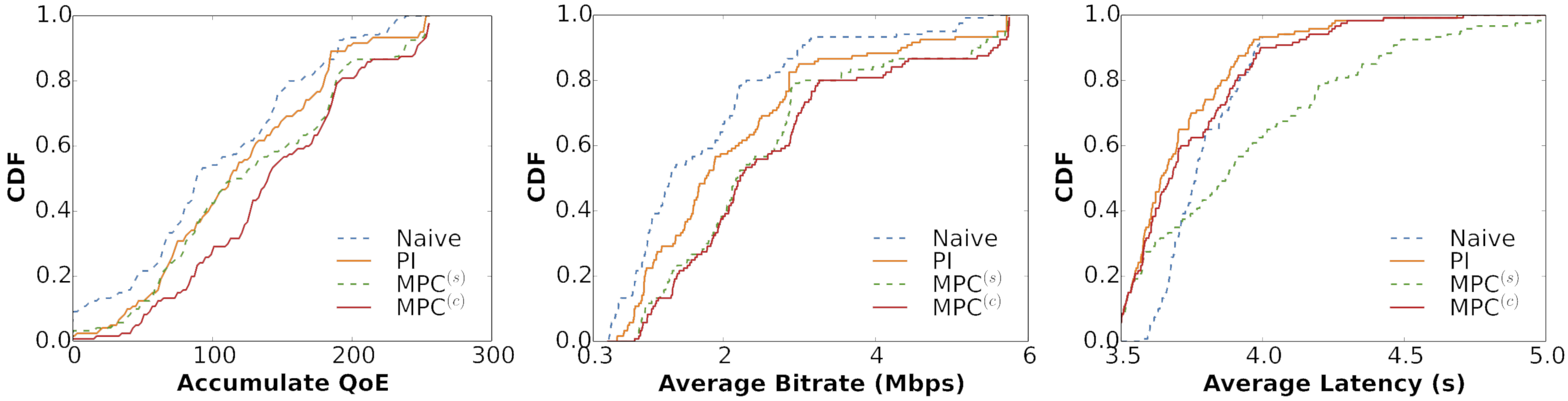


- Chunk Mode



Trace-driven Experiments and Evaluation

- 4G cellular bandwidth dataset with150 traces collected in NYC.
- Naïve ($\gamma\widehat{\omega}$), PI-Controller ($\gamma_p\widehat{\omega}$) and MPC (segment and chunk mode).



- MPC based algorithms outperform Naïve and PI-Controller.
- MPC^s suffers more latency (caused by freeze) than MPC^c.
- MPC^c achieves highest QoE with highest bitrate and lowest latency in most cases.

Conclusions & Ongoing Work

- Low latency is crucial, balance between latency and other QoE metrics
- MPC based streaming algorithms can improve the QoE performance with low latency.
- Chunk-based delivery is helpful to support low latency live video streaming.

- Optimal Streaming Policy from Deep Reinforcement Learning (DRL) vs. Model-based RL
- Optimal Playback Pace Adaption

THANK YOU!