互联网的核心应用：视频

- 用户观看视频的质量体验（QoE）直接影响视频内容提供商的收入，因此**提升用户的QoE至关重要**
- 提升QoE的根本挑战：**视频动态码率**与**网络带宽动态变化**之间的**不匹配性**

GLOBAL APPLICATION CATEGORY TRAFFIC SHARE

	Rank Change	Category	Downstream	Upstream
1	-	Video Streaming	48.9%	19.4%
2	-	Social Networking	19.3%	16.6%
3	2	Web	13.1%	23.1%
4	-1	Messaging	6.7%	20.4%
5	-	Gaming	4.3%	1.9%
6	-2	Marketplace	4.1%	1.2%
7	2	File Sharing	1.3%	6.6%
8	-1	Cloud	1.1%	6.7%
9	-3	VPN and Security	0.9%	3.9%
10	-	Audio	0.2%	0.2%

Sandvine 2021 Mobile Internet Phenomena Report

提升QoE的关键：ABR算法

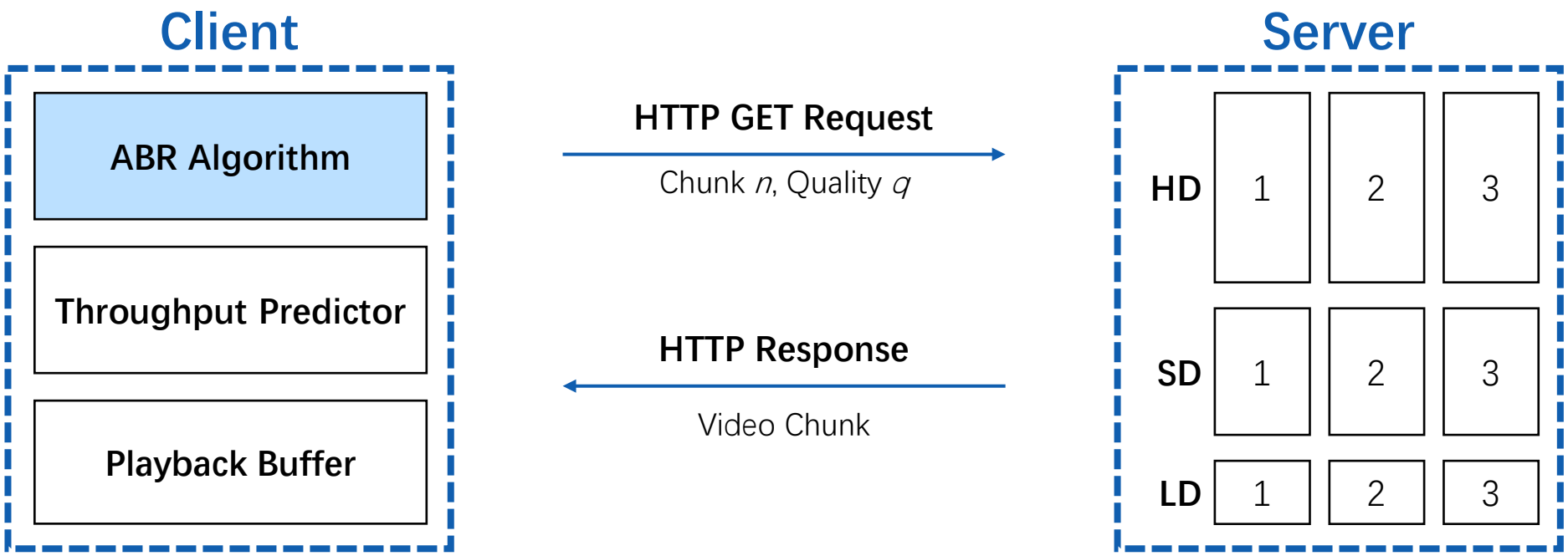
- 点播视频以**DASH** (Dynamic Adaptive Streaming over HTTP) 作为传输标准，已在业界广泛部署（YouTube, Netflix, Bilibili）
- 在DASH中，一个视频被编码为多种不同码率的版本，每个版本并被划分为多个等长的视频块（2~5s）
- DASH客户端运行**ABR** (Adaptive BitRate) 算法，为每一个视频块选择码率，目标是最大化**QoE** (Quality of Experience)

$$QoE = \sum_{k=1}^{N} \boldsymbol{q(R_k)} - \mu \sum_{k=1}^{N} \boldsymbol{\max\left(\left(\frac{d_k(R_k)}{T_k} - B_k\right), 0\right)} - \lambda \sum_{k=1}^{N-1} \boldsymbol{|q(R_{k+1}) - q(R_k)|}$$

视频质量　　卡顿时间　　质量切换

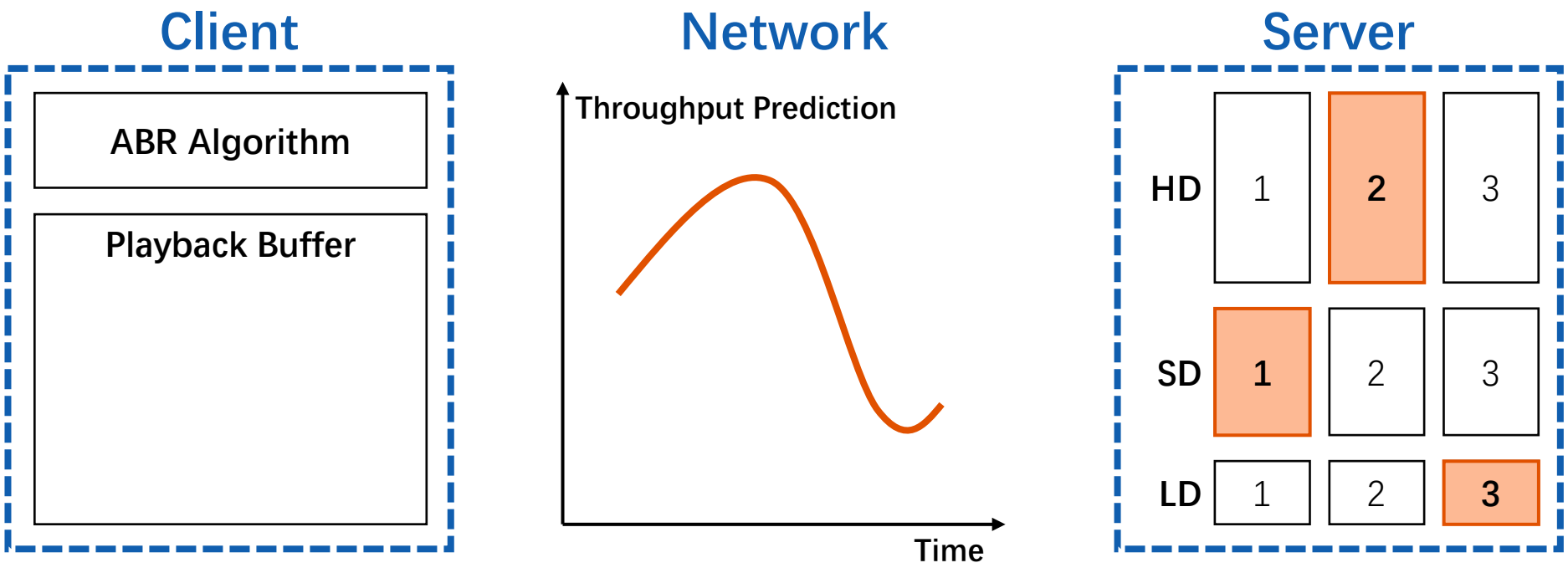
提升QoE的关键：ABR算法

- DASH客户端运行**ABR**算法，为每一个视频块选择码率，目标是最大化**QoE**
  - 目标：高质量，低卡顿，少质量切换
  - 输入：视频块吞吐量，播放缓冲区时长等
  - 输出：视频块的码率/清晰度



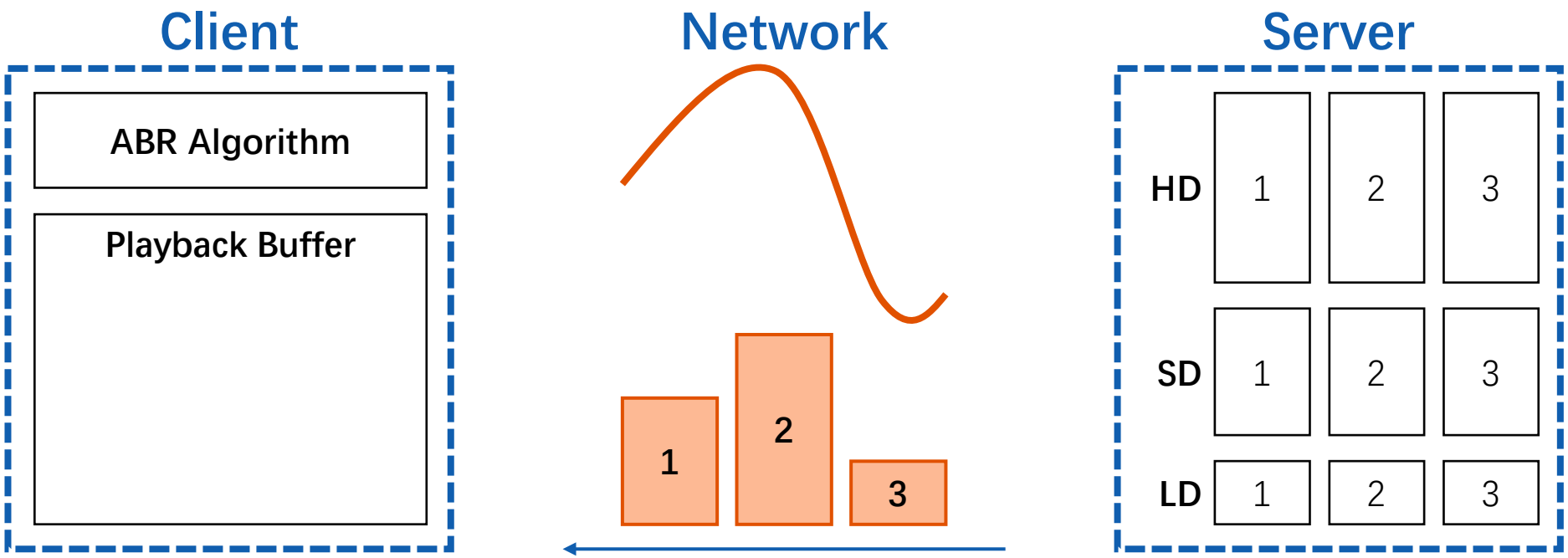
提升QoE的关键：ABR算法

- DASH客户端运行**ABR**算法，为每一个视频块选择码率，目标是最大化**QoE**
  - 目标：高质量，低卡顿，少质量切换
  - 输入：视频块吞吐量，播放缓冲区时长等
  - 输出：视频块的码率/清晰度



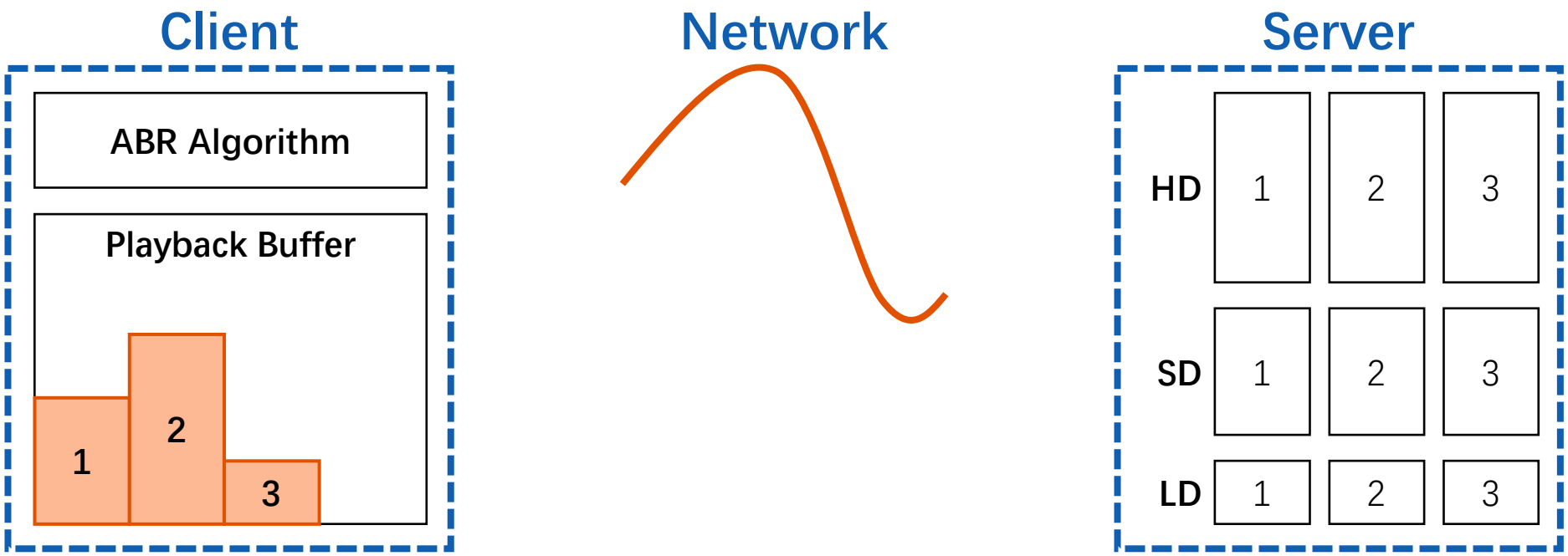
提升QoE的关键：ABR算法

- DASH客户端运行**ABR**算法，为每一个视频块选择码率，目标是最大化**QoE**
  - 目标：高质量，低卡顿，少质量切换
  - 输入：视频块吞吐量，播放缓冲区时长等
  - 输出：视频块的码率/清晰度



提升QoE的关键：ABR算法

- DASH客户端运行**ABR**算法，为每一个视频块选择码率，目标是最大化**QoE**
  - 目标：高质量，低卡顿，少质量切换
  - 输入：视频块吞吐量，播放缓冲区时长等
  - 输出：视频块的码率/清晰度



ABR算法学术研究

- 客户端ABR算法可分为四类
  - 决策逻辑：线性函数、控制论、机器学习等

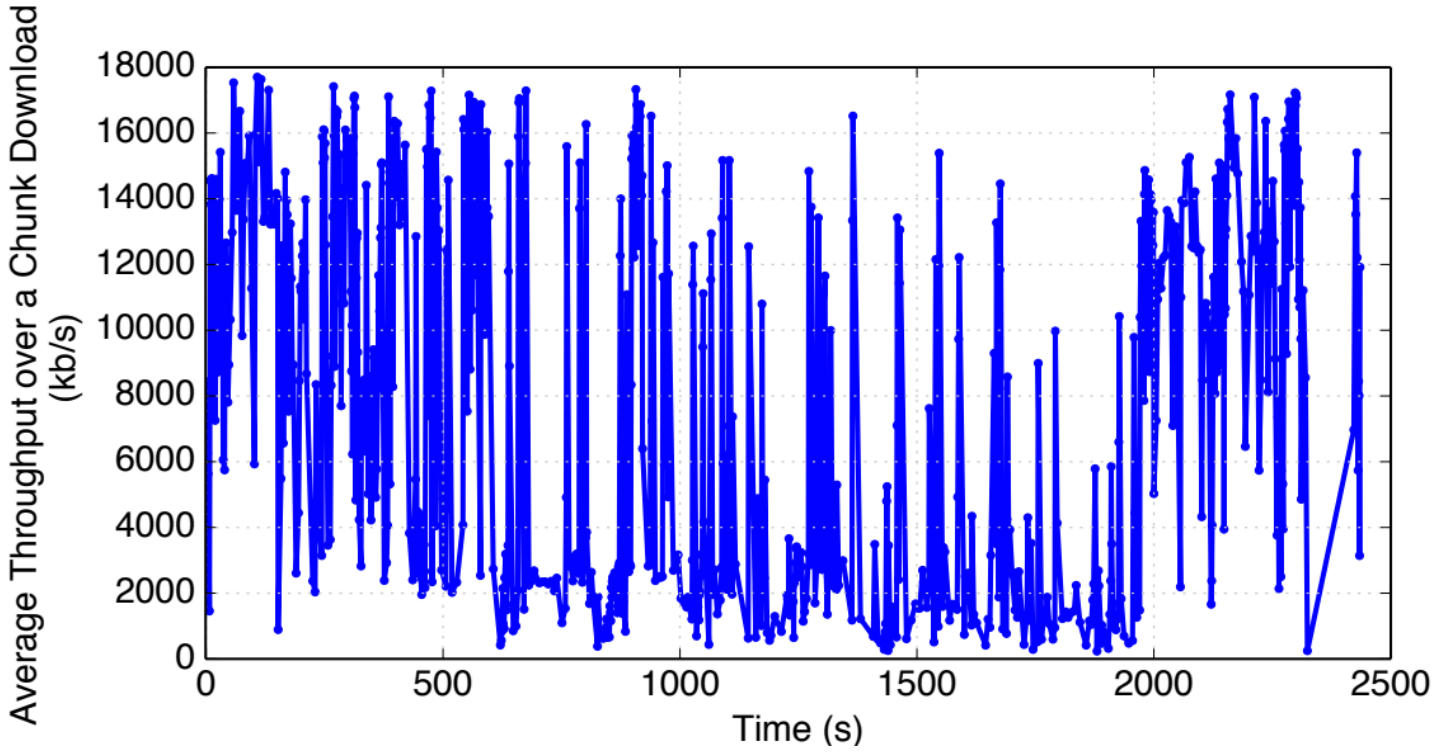
1 基于吞吐量	2 基于缓冲区	3 基于混合信息	4 基于机器学习
FESTIVE [CoNEXT'12] PANDA [JSAC'14] CS2P [SIGCOMM'16]	BBA [SIGCOMM'14] BOLA [INFOCOM'16]	MPC [SIGCOMM'15] PBA [HotMobile'15]	Pensieve [SIGCOMM'17] Comyco [MM'19]

BBA [SIGCOMM'14]

A Buffer-Based Approach to Rate Adaptation: Evidence from a Large Video Streaming Service

研究动机

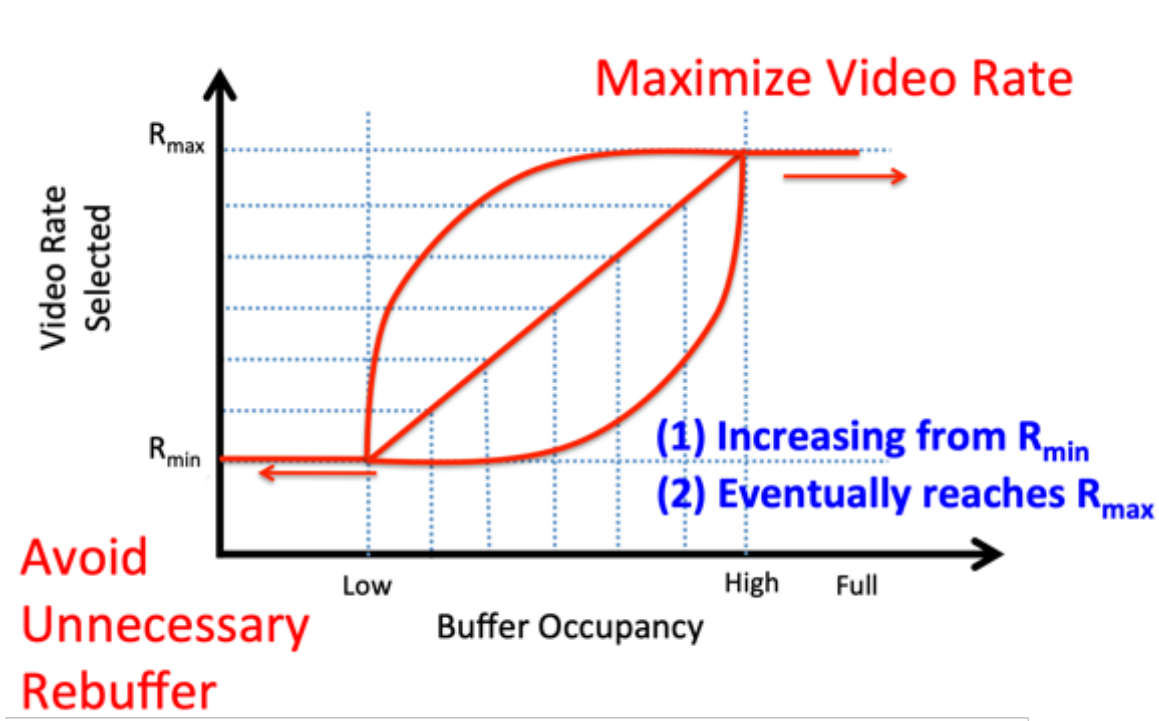
- 基于吞吐量的ABR算法依赖于吞吐量预测
- **挑战**：应用层（HTTP）吞吐量高度动态变化，难以准确预测
  - 受竞争流量、网络环境等因素影响



Basic idea：避免预测吞吐量，仅**基于缓冲区水平**选择码率

Buffer-Based Algorithm

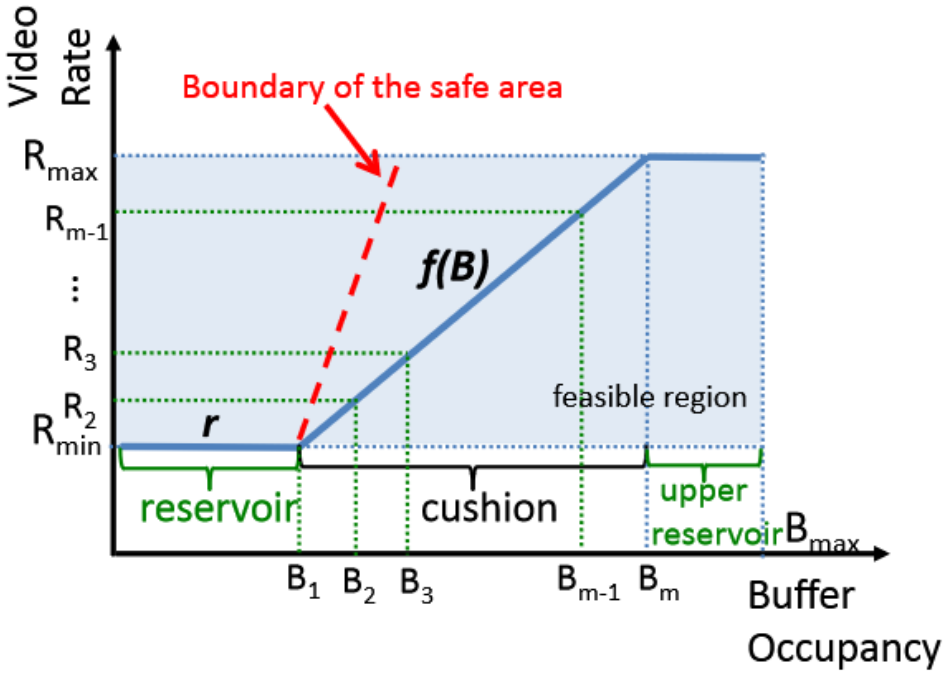
- 思想：在稳定阶段，将码率选择视作缓冲区水平的函数
  - 缓冲区水平隐含了关于网络容量变化的信息
  - 在启动阶段，依然需要基于吞吐量预测选择码率



线性函数：平衡视频**码率**与**卡顿**

BBA-0决策逻辑

- 思想：在稳定阶段，将码率选择视作缓冲区水平的**线性函数**
- 设计：两个缓冲区阈值，reservoir和cushion
  - Buffer Occupancy < reservoir：选择最低码率
  - Buffer Occupancy > reservoir + cushion：选择最高码率
  - Otherwise：通过线性函数确定缓冲区水平对应的码率级别



BBA变体

BBA-0 → **BBA-1** → **BBA-2**

BBA-0	BBA-1	BBA-2
针对固定码率（CBR）视频 建立码率与缓冲区水平的线性函数	支持可变码率（VBR）视频 将码率映射替换为视频块大小映射	基于吞吐量预测 支持在启动阶段选择码率

参考资料：https://blog.csdn.net/LvGreat/article/details/131211119?spm=1001.2014.3001.5501

实验评估设置

❖ 实验环境：Netflix

- 将BBA部署于基于浏览器的播放器中
- 2013年9月，一周50W用户

❖ 对比方法

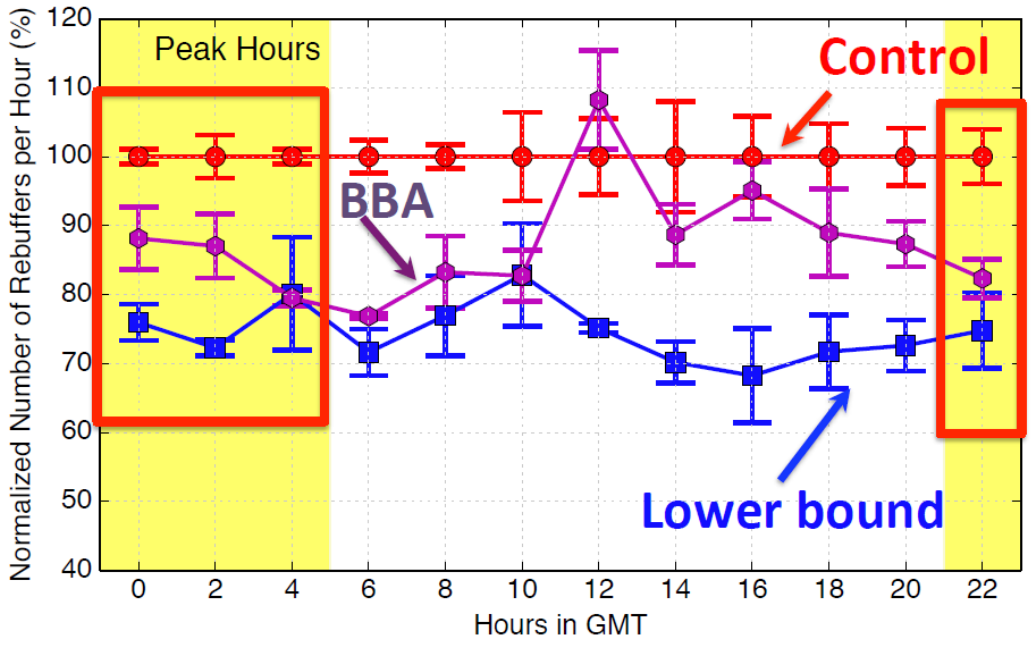
- Control：Netflix默认算法
- Lower bound：持续选择最低码率（代表卡顿的下限）

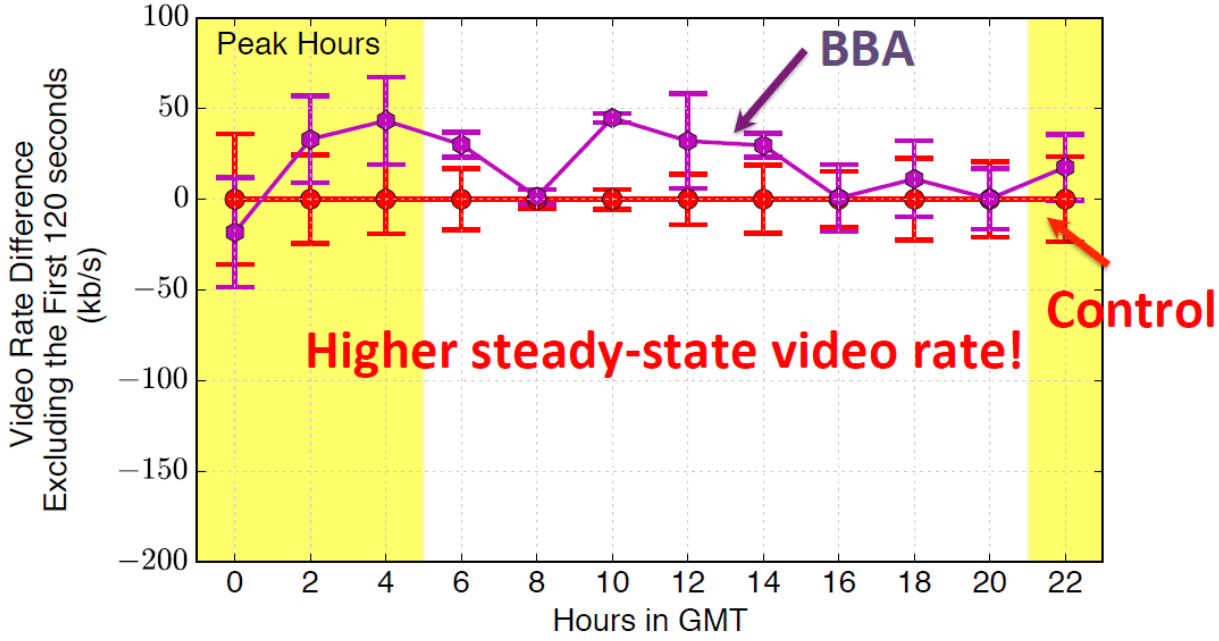
❖ 性能指标

- 卡顿
- 码率

BBA-2评估结果

- 效果：BBA-2实现了10%-20%的卡顿下降，同时也提高了稳定阶段的码率





总结

❖ 优势&研究亮点

- 首个仅基于缓冲区进行码率选择的算法
- BBA-0设计与实现简单，便于部署
- BBA-0在真实场景中码率和卡顿表现优秀（Learning in Situ [NSDI'20]）

❖ 不足&后续研究

- 在稳定阶段完全忽略了吞吐量信息 → MPC [SIGCOMM'15]
- 码率切换较为频繁 → BBA-Others
- 线性映射未必最优 → BOLA [INFOCOM'16] / Stick [INFOCOM'21]
- 缺乏明确的QoE优化目标 → MPC [SIGCOMM'15] / BOLA [INFOCOM'16]

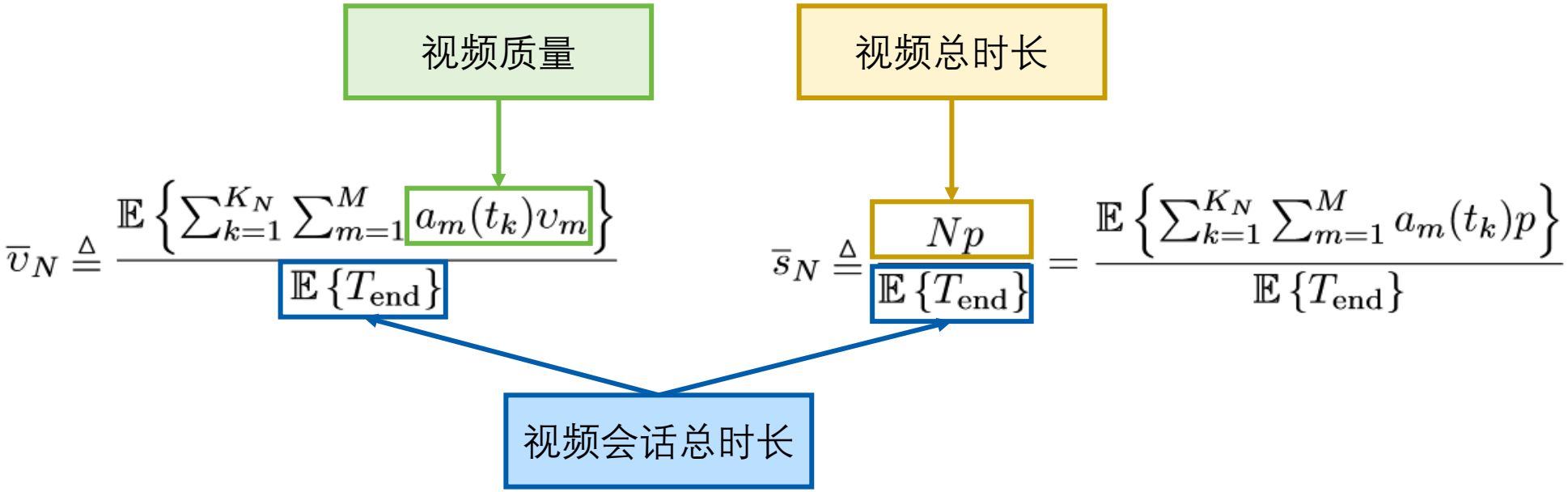
BOLA [INFOCOM'16]

BOLA: Near-optimal bitrate adaptation for online videos

BOLA

❖ 思想：将ABR形式化为**效用最大化问题**，基于**李雅普诺夫优化**求解

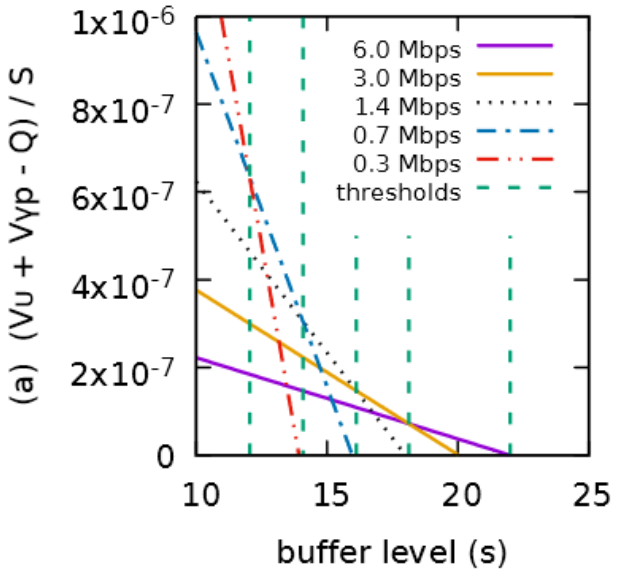
- 两类效用：提高平均**视频质量**$\bar{v}_N$，降低卡顿时间（即增加**不卡顿时间**占比 $\bar{s}_N$）
- **总效用** $= \bar{v}_N + \gamma \bar{s}_N$

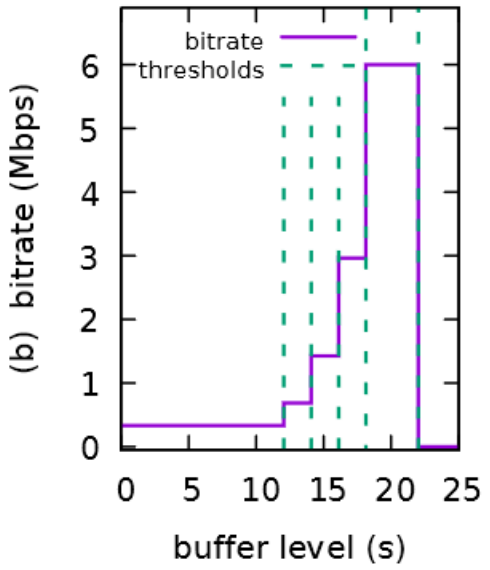


BOLA-BASIC决策逻辑

- 逻辑：求解$\left(V v_m + V\gamma p - Q(t_k)\right)/S_m$的最大值，为下一视频块选择码率
  - V、γ：调整参数
  - v_m：码率级别m的视频质量（与码率呈对数关系）
  - p：视频块时长
  - $Q(t_k)$：第k个时隙中缓冲区的视频块数量
  - S_m：码率级别m的视频块大小

- 决策函数：**阶梯状**





BOLA变体

参考资料：https://blog.csdn.net/LvGreat/article/details/130487156?spm=1001.2014.3001.5501

实验评估设置

❖ **实验环境**：Trace-driven Simulation

❖ **对比方法**

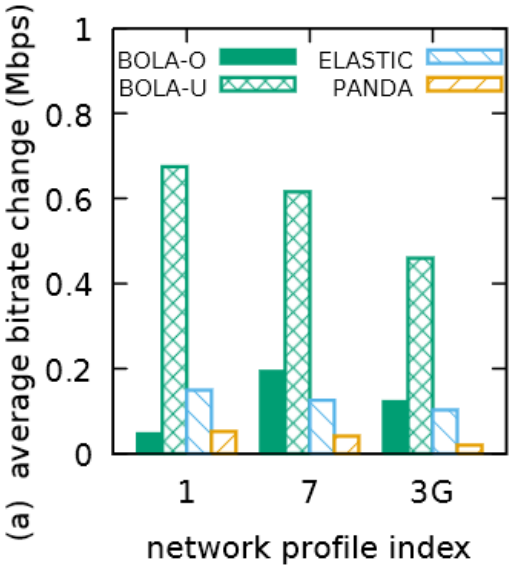
- ELASTIC、PANDA
- 离线最优（基于动态规划，代表最高效用）

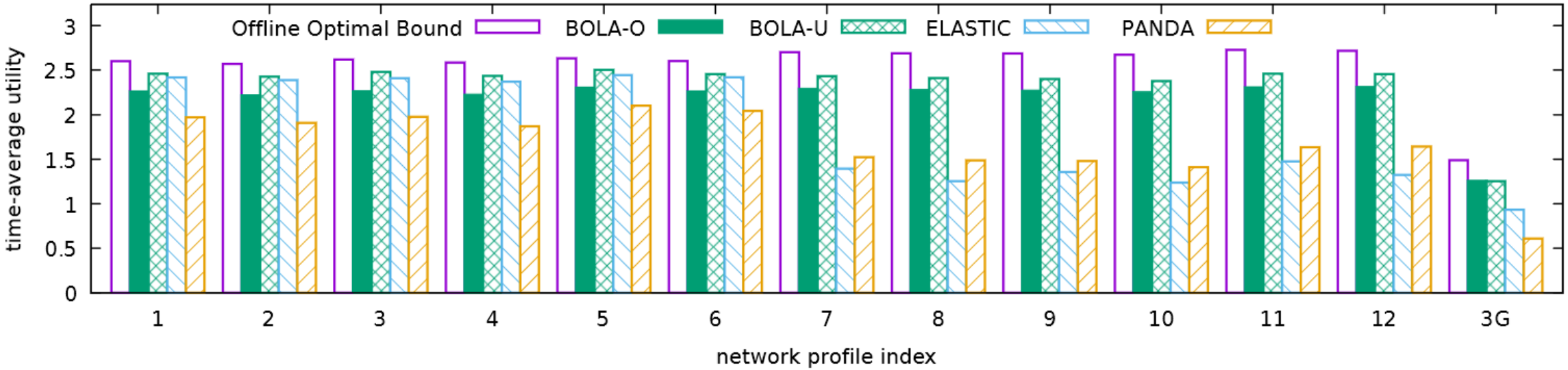
❖ **数据集**

- 网络trace：Norway 3G/HSDPA、DASH benchmarks
- 视频：Big Buck Bunny（10min），视频块大小3s，10种码率级别，缓冲区上限25s

BOLA-O & BOLA-U评估结果

- 效果：
  - BOLA-U的效用始终优于对比算法，且在离线最优的84%-95%范围内
  - BOLA-O的效用略低于BOLA-U，但码率切换程度明显降低





总结

❖ 优势&研究亮点

- 基于缓冲区的ABR算法代表作之一
- BOLA-O作为dash.js开源视频播放器默认ABR算法的核心部分，在真实场景中得以应用
- 基于李雅普诺夫优化方法设计，理论性较强

❖ 不足&后续研究

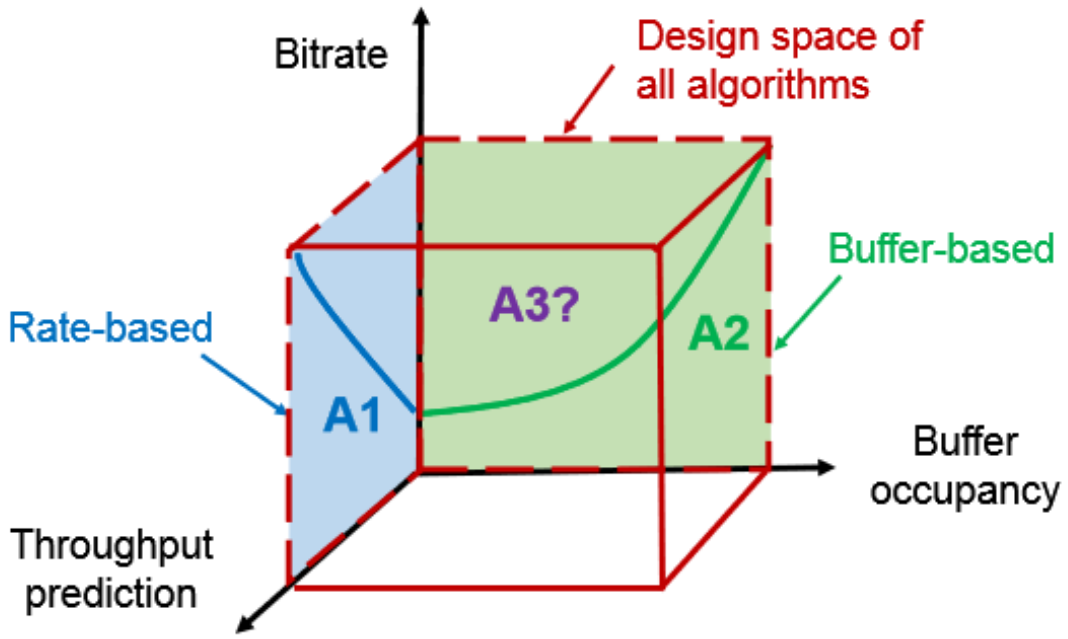
- 稳定阶段忽略吞吐量信息 → MPC [SIGCOMM'15] / BOLA-E & DYNAMIC [MMSys'18]
- 缺乏统一的QoE优化目标 → MPC [SIGCOMM'15]
- 不同环境下的最优参数难以确定 → Oboe [SIGCOMM'18]
- 除了ABR算法外还涉及视频块请求等逻辑，实现繁琐且偏工程化，部署较为困难

MPC [SIGCOMM'15]

A Control-Theoretic Approach for Dynamic Adaptive Video Streaming over HTTP

研究动机

- 设计空间：过往算法或只基于吞吐量，或只基于缓冲区水平



Basic idea：将**吞吐量**与**缓冲区**信息结合以进行码率选择

Model Predictive Control

- 思想：将ABR形式化为**随机最优控制**问题，利用**模型预测控制（MPC）**求解

- 设计选择： strawman solutions
  - 比例-积分-微分（PID）：目标是稳定性而非优化，且适用于连续系统而非离散系统
  - 马尔可夫决策过程（MDP）：假设吞吐量变化服从马尔可夫过程，实践中不一定否成立
  - **模型预测控制（MPC）**：将码率选择视为有限时域内的QoE最大化问题，适用于ABR算法

Model Predictive Control

❖ 原理：预测关键变量，求解优化问题

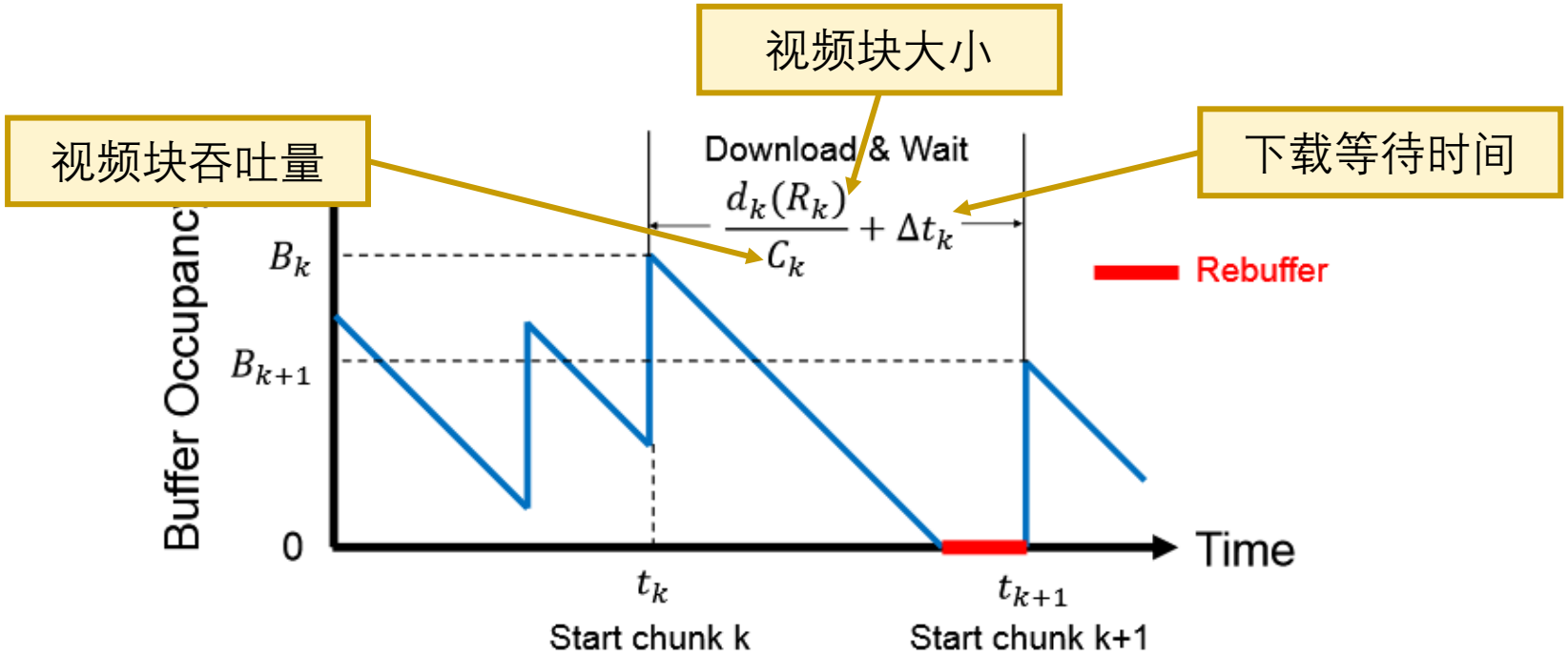
- 预测有限时间范围的吞吐量
- 基于预测吞吐量，根据**视频缓冲区模型**，估计解空间中每组解取得的QoE
- 选择能够使总体**QoE目标**最大的一组码率选择作为解

❖ 算法步骤：

- 解空间初始化：考虑未来5个视频块的6种码率选择，解空间大小为6^5
- 预测吞吐量：计算过去5个视频块的调和平均数，作为吞吐量预测值
- 最大化QoE：遍历解空间中的每组解（即未来5个视频块的码率选择），计算每组解对应的预期QoE，记录最大值
- 输出决策：将能够使QoE最大化的一组解的第一个码率选择作为算法输出

视频缓冲区模型

- 缓冲区水平变化
  - 减少：视频观看过程中，缓冲区水平随播放时间匀速减少
  - 增加：视频块下载完成后，缓冲区水平增加一个视频块的时长
- 卡顿：初始缓冲区水平 < 视频块下载时间（忽略下载等待时间）
  - 视频块下载时间 = 视频块大小$d_k(R_k)$ / 视频块吞吐量C_k



QoE目标

❖ 将QoE目标建模为多个指标的**线性函数**

- 视频质量：$q(R_k)$ （可简化为码率：R_k）
- 卡顿时间：$\max\left(\left(\frac{d_k(R_k)}{T_k}-B_k\right),0\right)$
- 质量切换：$|q(R_{k+1})-q(R_k)|$ （可简化为码率切换：$|R_{k+1}-R_k|$）
- 启动延迟：T_s （通常忽略）

$$QoE=\sum_{k=1}^{N}\boldsymbol{q(R_k)}-\mu\sum_{k=1}^{N}\boldsymbol{\max\left(\left(\frac{d_k(R_k)}{T_k}-B_k\right),0\right)}-\lambda\sum_{k=1}^{N-1}\boldsymbol{|q(R_{k+1})-q(R_k)|}-\mu_s\boldsymbol{T_s}$$

QoE最大化问题

❖ 参数

- $\mu, \lambda, \mu_s, q(\cdot)$

❖ 输入

- 视频块时长：L
- 视频块大小：$d_k(\cdot)$
- 缓冲区上限：B_{max}
- 吞吐量trace：$\{C_t\}$

需要预测

❖ 输出

- 码率决策：$R_1, \cdots, R_K$（K为最大视频块序号）
- 启动延迟：T_s
- 下载等待时间：Δt_k

$$\max_{R_1,\cdots,R_K,T_s} \quad QoE_1^K \tag{6}$$

$$s.t. \qquad t_{k+1} = t_k + \frac{d_k(R_k)}{C_k} + \Delta t_k, \tag{7}$$

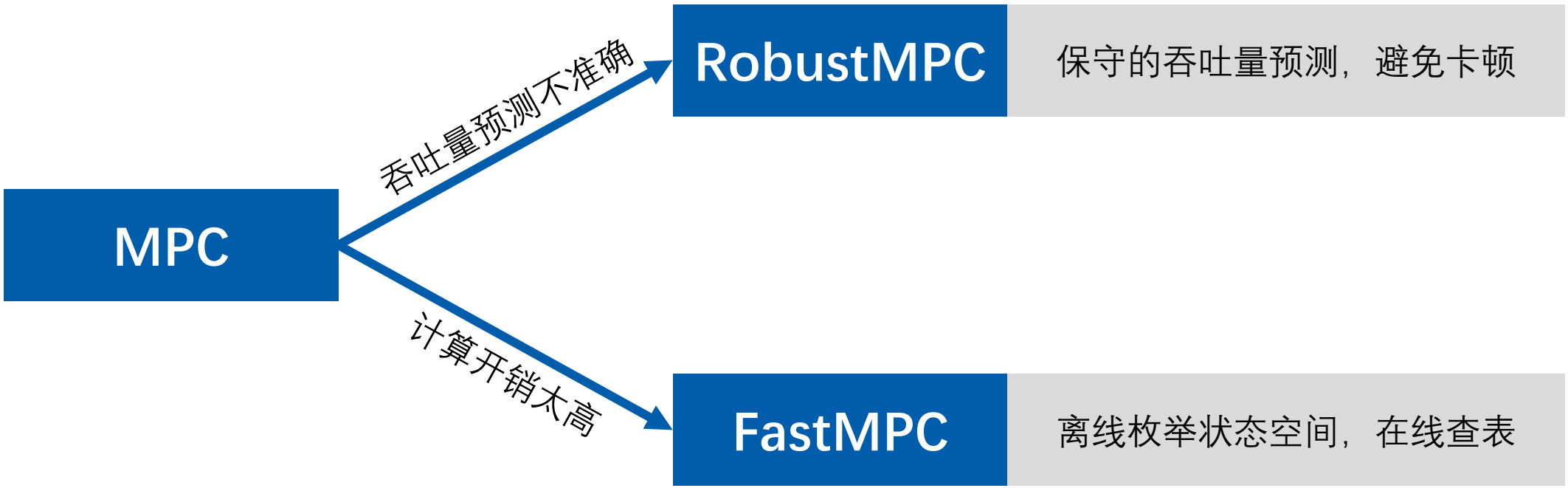
$$C_k = \frac{1}{t_{k+1} - t_k - \Delta t_k} \int_{t_k}^{t_{k+1}-\Delta t_k} C_t \, dt, \tag{8}$$

$$B_{k+1} = \left(\left(B_k - \frac{d_k(R_k)}{C_k} \right)_+ + L - \Delta t_k \right)_+, \tag{9}$$

$$B_1 = T_s, \quad B_k \in [0, B_{max}] \tag{10}$$

$$R_k \in \mathcal{R}, \quad \forall k = 1, \cdots, K. \tag{11}$$

MPC变体


MPC
吞吐量预测不准确
计算开销太高
RobustMPC
保守的吞吐量预测，避免卡顿
FastMPC
离线枚举状态空间，在线查表

实验评估设置

❖ 实验环境：Trace-driven仿真

- Emulation（性能评估）：Linux tc限速，基于dash.js播放器进行实验
- Simulation（敏感性分析）

❖ 对比方法

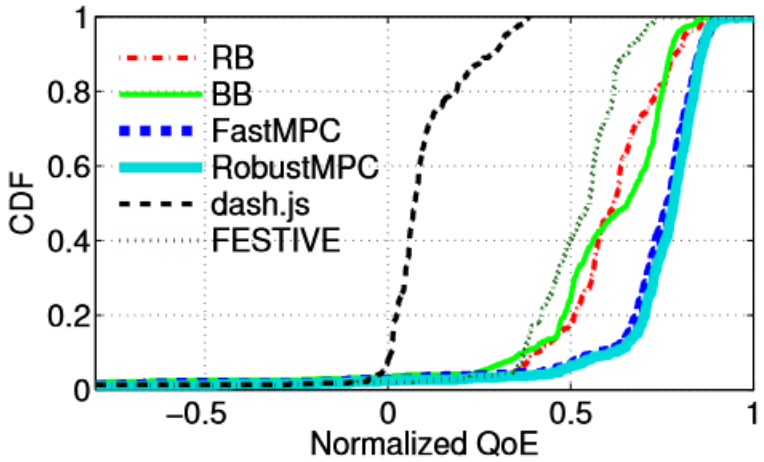
- RB、BB（BBA-0）、FESTIVE [CoNEXT'12]、dash.js默认类RB算法

❖ 数据集

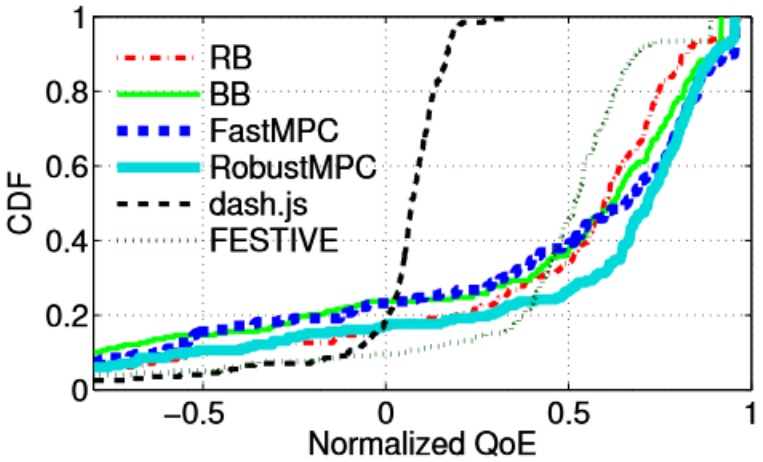
- 网络trace：FCC、Norway 3G/HSDPA、Synthetic
- 视频：Envivio（260s），视频块大小4s，码率级别{350kbps, 600kbps, 1000kbps, 2000kbps, 3000kbps}，缓冲区上限30s

RobustMPC & FastMPC评估结果

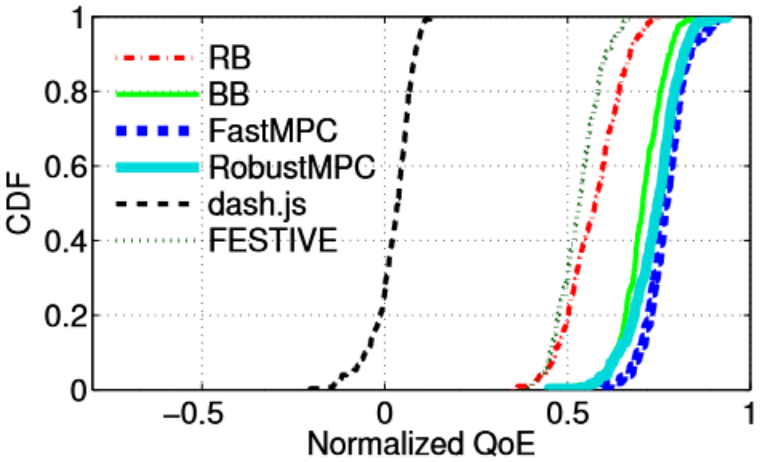
- 效果：
  - RobustMPC在三种数据集上相对次优算法分别实现了15%、10%和5%的QoE提升
  - FastMPC的性能受预测错误率影响，在FCC和Synthetic数据集上表现较好，在HSDPA数据集上性能有所下降



(a) FCC dataset



(b) HSDPA dataset



(c) Synthetic dataset

总结

❖ 优势&研究亮点

- 基于混合信息的ABR算法代表作，扩展了ABR算法的设计空间
- 基于控制论方法设计，性能有理论保证
- MPC与RobustMPC实现较为简单，便于部署
- 统一了不同的QoE优化目标，是后续研究工作的基础

❖ 不足&后续研究

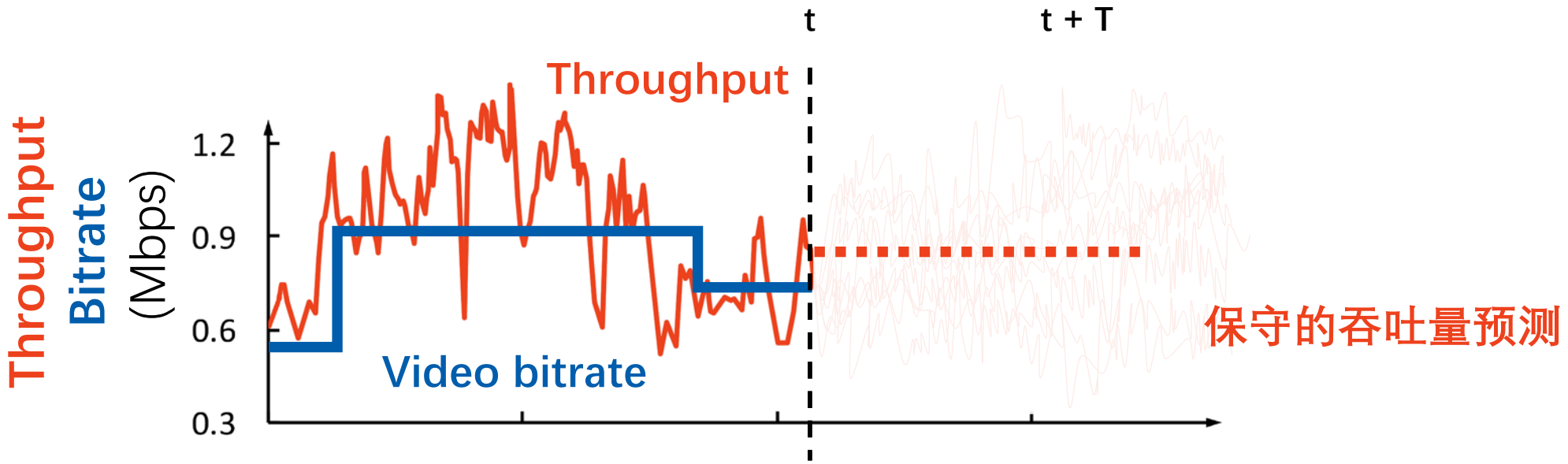
- 依赖准确的吞吐量预测 → CS2P [SIGCOMM'16] / Fugu [NSDI'20] / Lumos [INFOCOM'22]
- MPC计算开销较大 → PiTree [MM'19]
- 线性QoE不一定符合真实用户感知 → ITU-P P.1203

Pensieve [SIGCOMM'17]

Neural Adaptive Video Streaming with Pensieve

研究动机

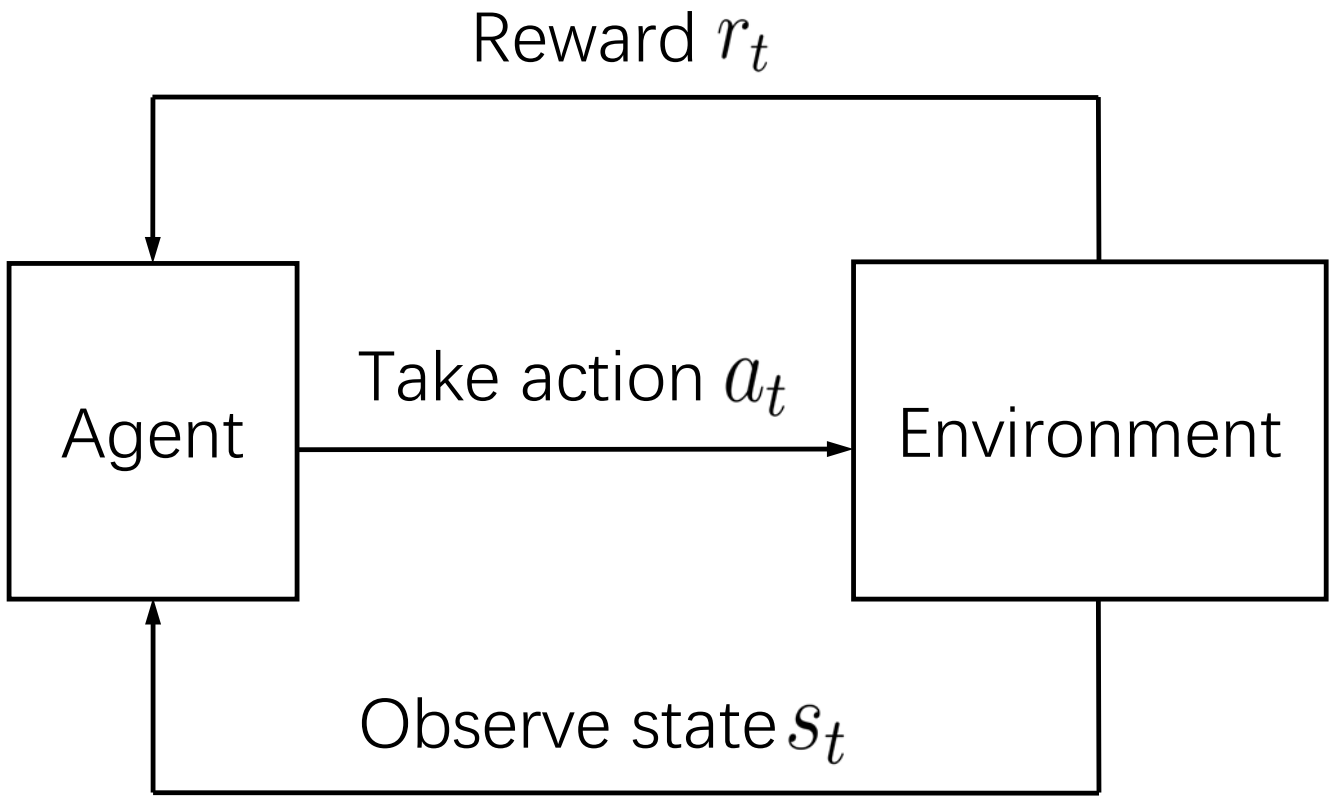
- MPC的性能依赖于准确的吞吐量预测，在高动态网络环境下很难实现
- RobustMPC采用更保守的吞吐量预测，但很难实现最优性能



Basic idea：从视频流会话中**学习**网络环境变化

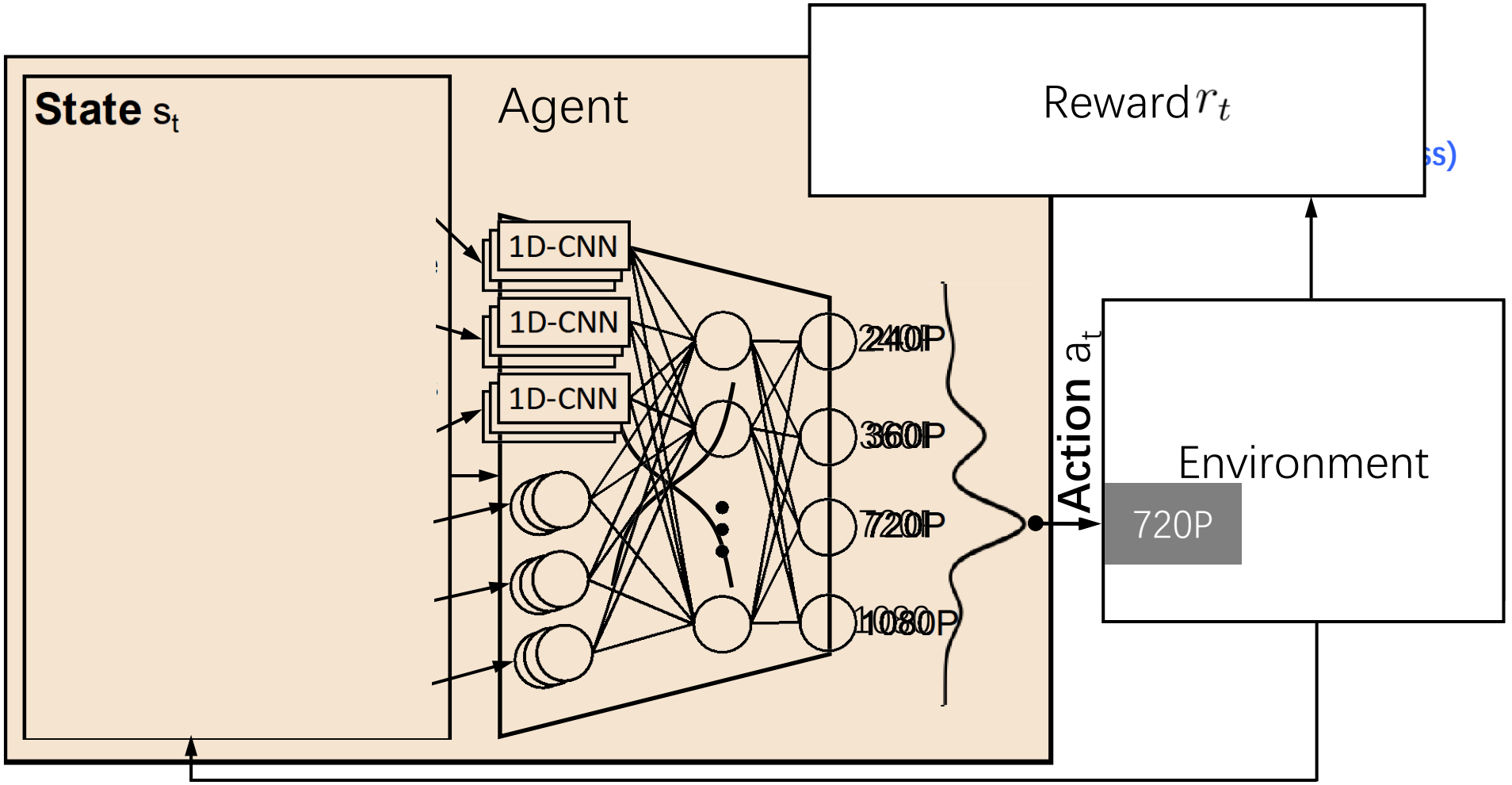
Pensieve原理：强化学习

- 框架：代理（Agent）+环境（Environment）
  - 代理基于观察的状态（输出特征），做出行动（输出码率决策），获得奖励（QoE分数）

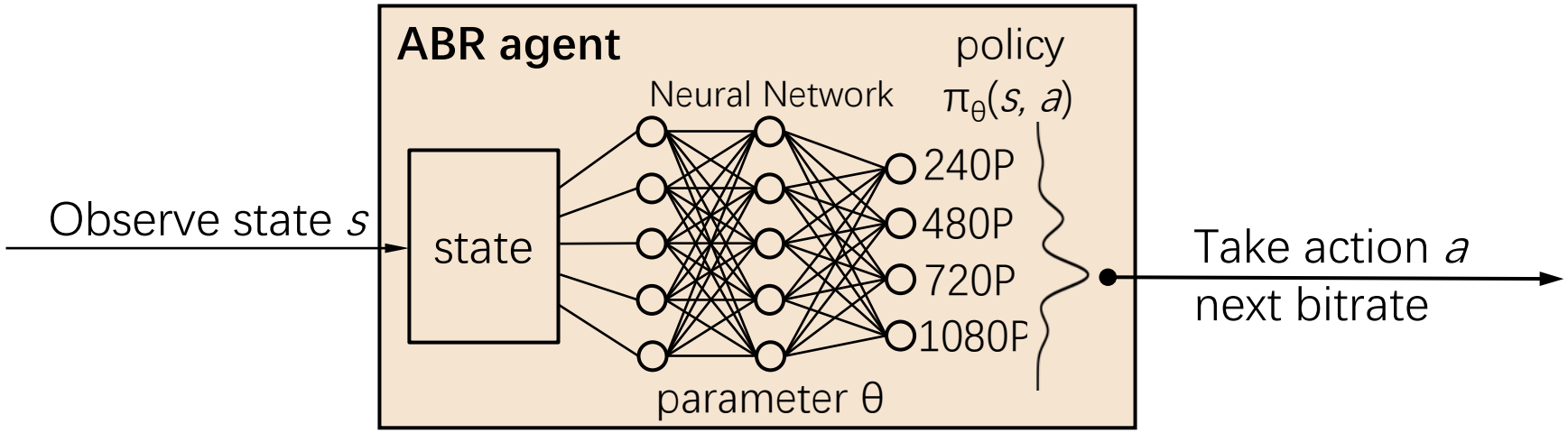


目标：最大化累积奖励 $\sum_t r_t$

Pensieve设计


State s_t
Agent
Reward r_t
1D-CNN
1D-CNN
1D-CNN
240P
360P
720P
1080P
Action a_t
Environment
720P

Pensieve训练



Collect experience data: trajectory of [state, action, reward]

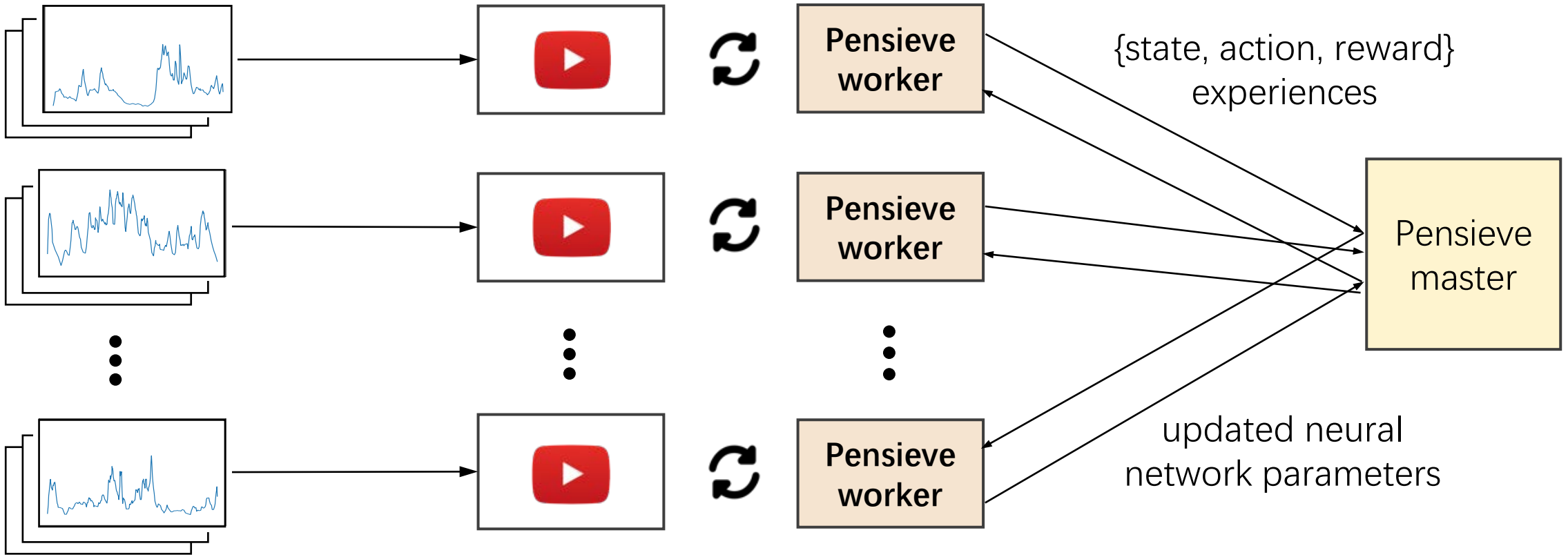
Training: $\theta \leftarrow \theta + \alpha \nabla_\theta \mathbb{E}_{\pi_\theta} \left[\sum_t r_t \right]$ **estimate from empirical data**

Pensieve训练系统（A3C）

Large corpus of network traces
cellular, broadband, synthetic

Video playback
Fast chunk-level simulator

Model update
TensorFlow



实验评估设置

❖ **实验环境**：Trace-driven Emulation & Simulation

❖ **对比方法**

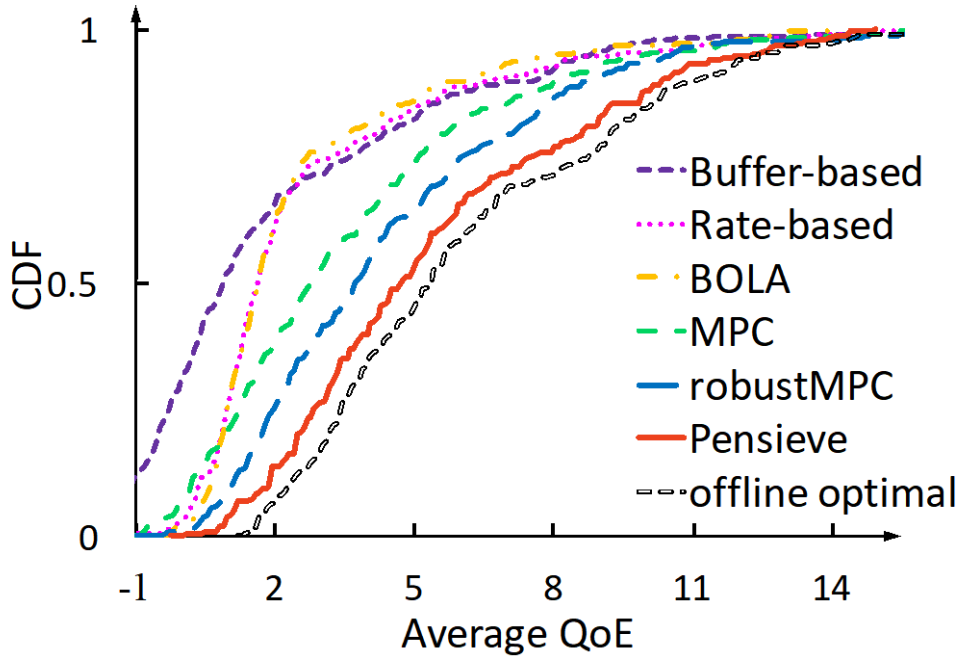
- RB、BB（BBA-0）、BOLA、MPC、RobustMPC
- 离线最优（代表性能上限）

❖ **数据集**

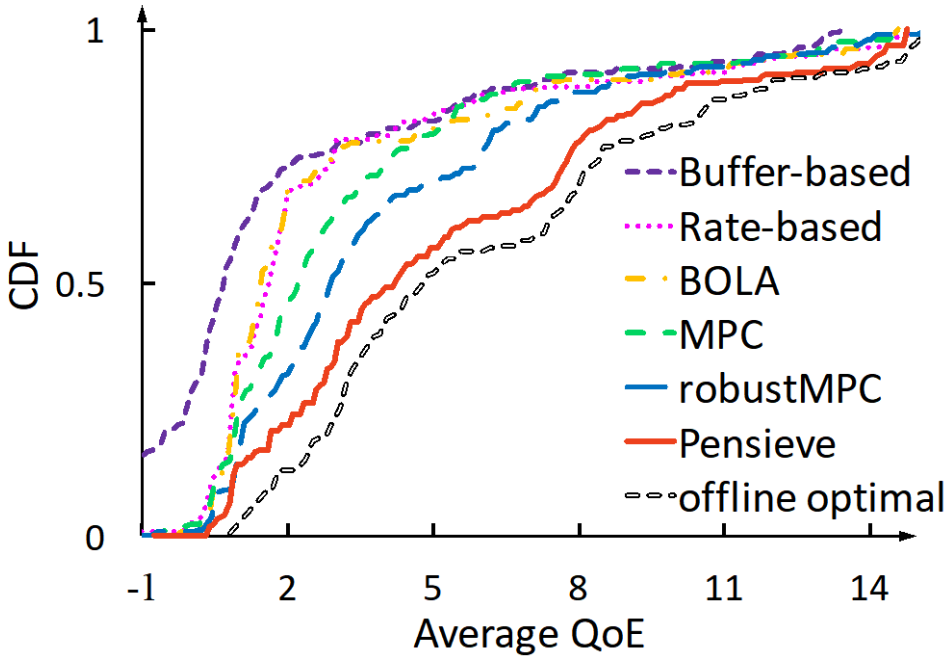
- 网络trace：FCC、Norway 3G/HSDPA
- 视频：EnvivioDash3（193s），视频块大小4s，码率级别{300, 750, 1200, 1850, 2850, 4300} kbps，缓冲区上限60s

Pensieve评估结果

- 总体QoE:
  - Pensieve相对于次优算法取得了12%-25%的QoE提升
  - Pensieve与离线最优的性能差距在9%-14%以内



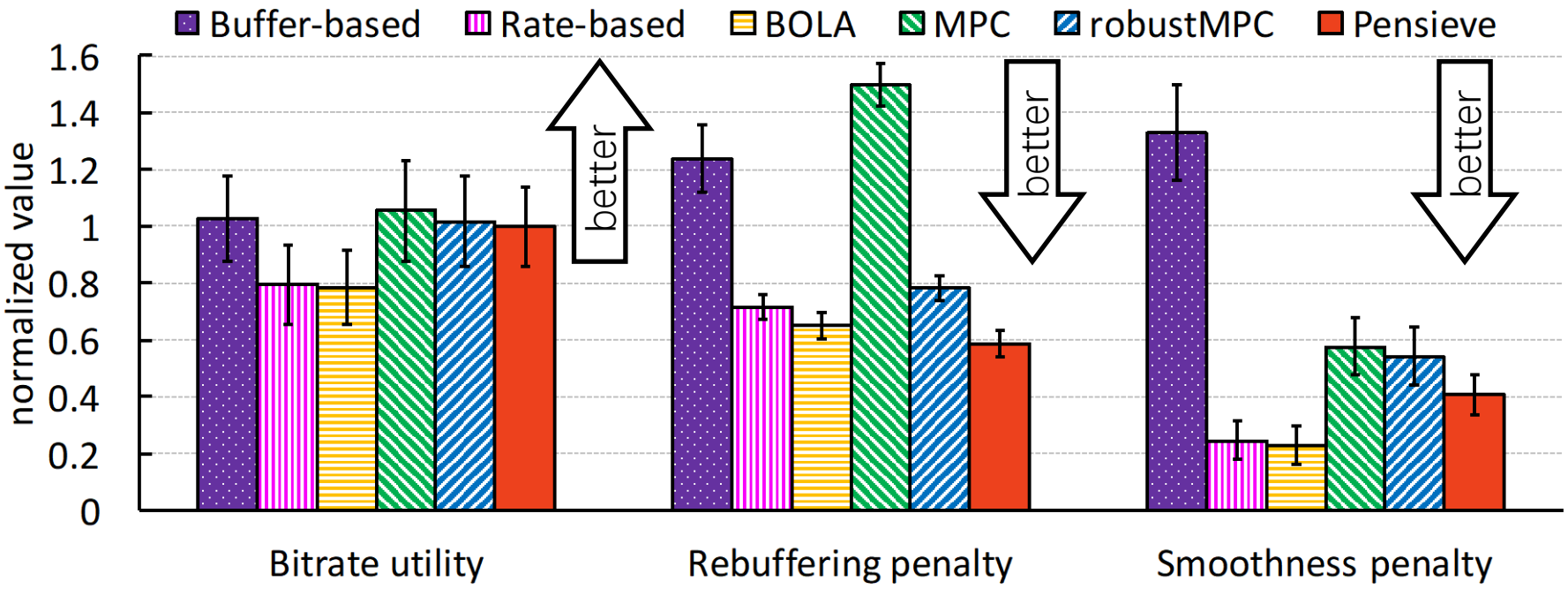
Norway 3G cellular dataset



FCC broadband dataset

Pensieve评估结果

- QoE细分指标：
  - Pensieve相对于次优算法降低了10%-32%的卡顿



总结

❖ 优势&研究亮点

- 基于机器学习类的ABR算法代表作
- 成功将强化学习方法应用于ABR算法中，工作开源，为后续研究奠定基础

❖ 不足&后续研究

- 机器学习的问题（轻量化与可解释性等） → PiTree [MM'19] / Metis [SIGCOMM'20]
- 离线仿真环境与真实网络系统存在偏差 → CausalSim [NSDI'23]
- 其他：真实环境部署（ABRL [ICML'19 RL4RealLife Workshop]）、模仿学习（Comyco [MM'19]）、在线学习（OnRL [MobiCom'20]）、长尾学习与泛化（A2BR [JSAC'22]）等