End-to-end Congestion Control as Learning for Unknown Games with Bandit Feedback

Zhiming Huang (UVic), Kaiyang Liu (MUN), and Jianping Pan (UVic)

End-to-end Congestion Control

- Network congestion may occur when a sender overflows the network with too many packets
- End-to-end congestion control relies on limited information, e.g., round-trip time (RTT), packet loss rate.

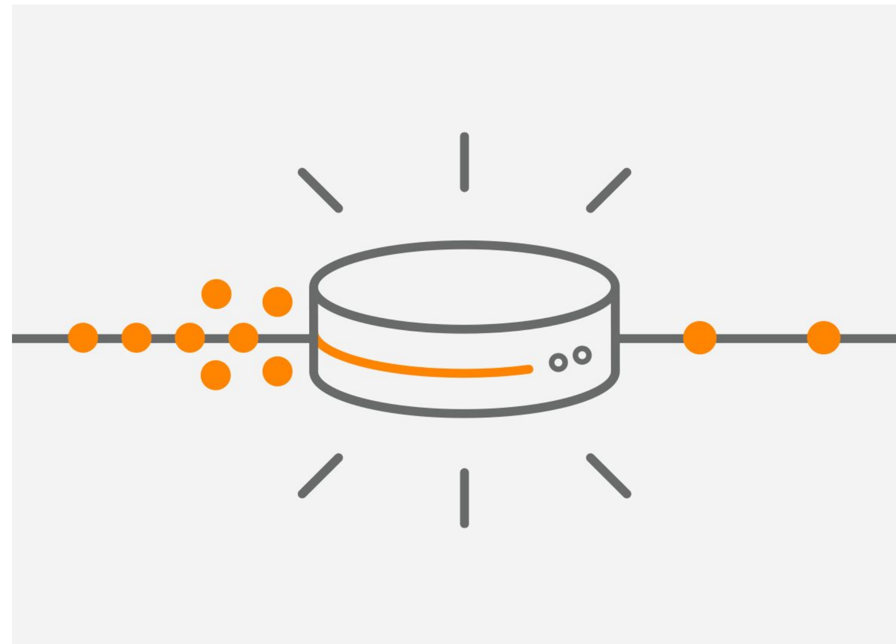


Figure source: https://www.noction.com/blog/tcp-transmission-control-protocol-congestion-control

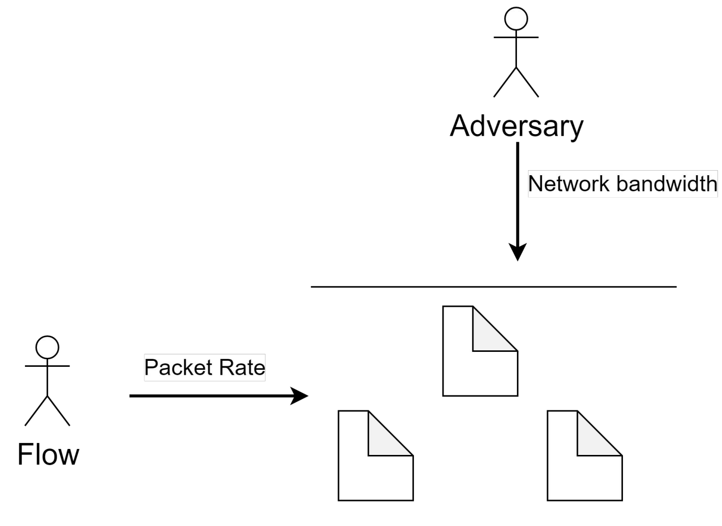
Contributions

- Take a step forward to the open problems raised by Karp et al. in FOCS 2000 to have a further understanding of the competition nature of end-to-end congestion control

- Swap-regret-minimization as a design concept or building block for congestion control algorithms

Two-agent Game Models [Karp et al., 2000]

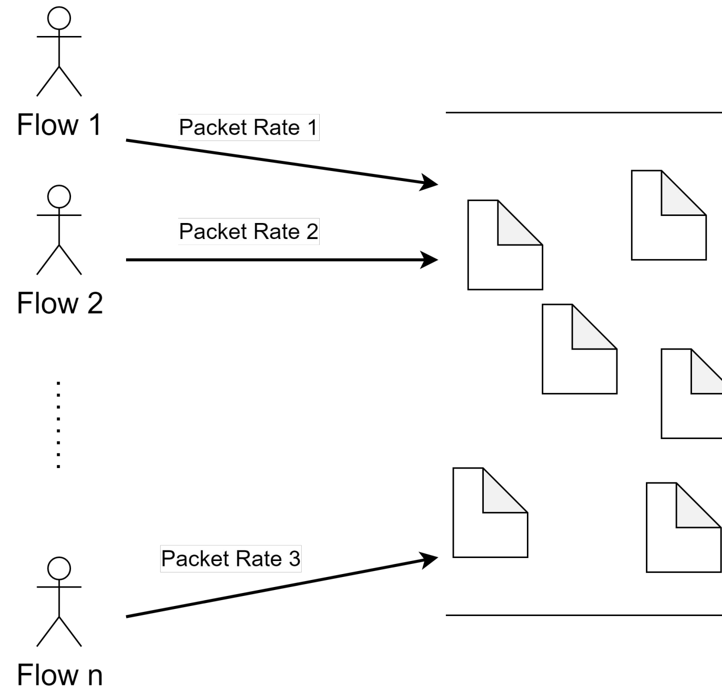
- Game is repeated for T rounds
- Flow decides packet sending rate
- Adversary decides the available network bandwidth (not revealed to the flow)
  - Static case: fixed over time
  - Dynamic case: change over time, even adaptively
- Then, the flow received a utility as a result of the packet sending rate and available network bandwidth



Open Problems Raised by Karp et al. (2000)

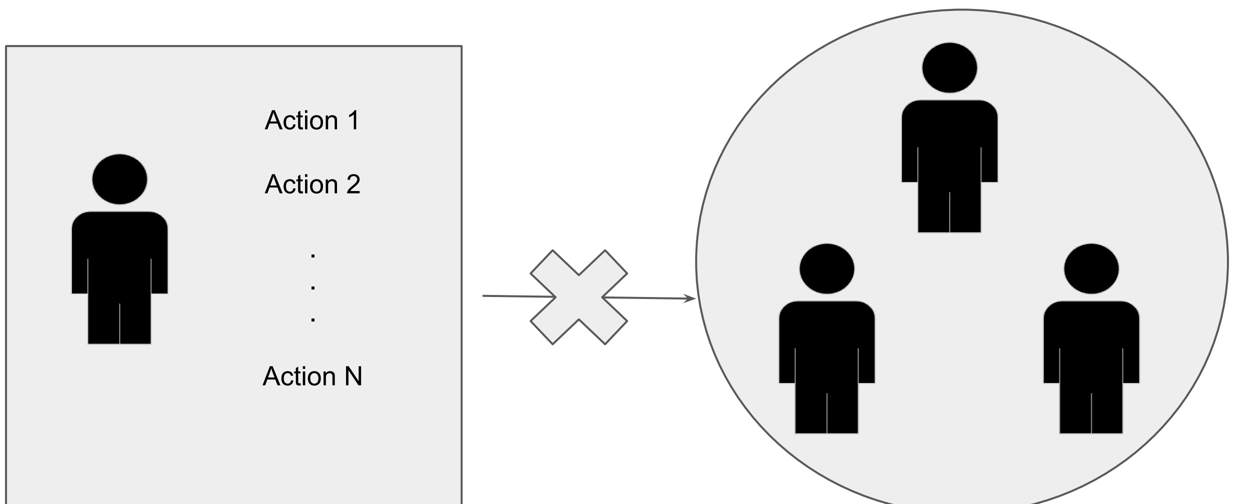
- The dynamic of the available network bandwidth is a joint result of the competition among multiple flows

- Randomized algorithms to address the dynamic network bandwidth



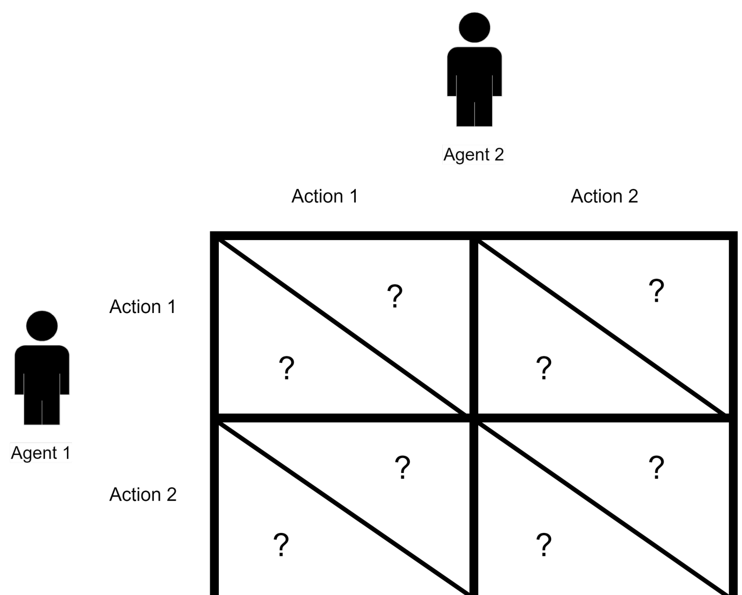
A Step Forward: Unknown Games with Bandit Feedback

- Unknown games (black-box games):
  - Each agent does not know the number, actions, and feedback of other agents



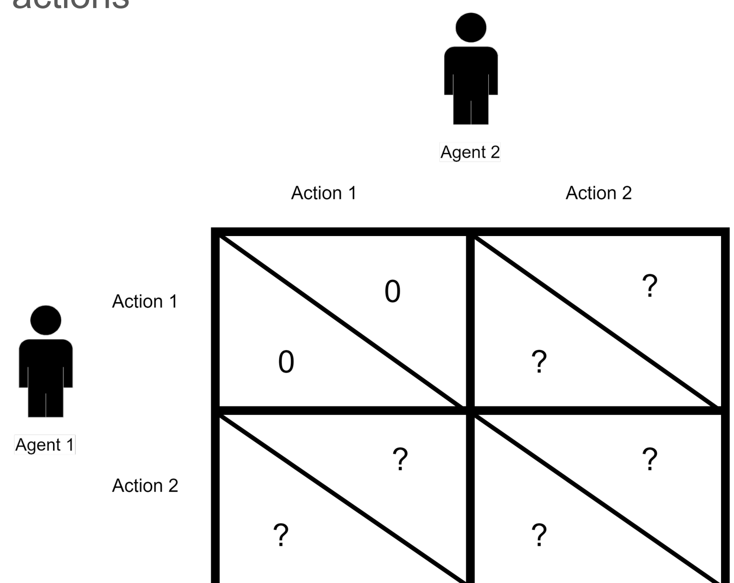
A Step Forward: Unknown Games with Bandit Feedback

- Unknown games (black-box games):
  - Each agent does not know the number, actions, and feedback of other agents
  - Each agent does not know the underlying game structure



A Step Forward: Unknown Games with Bandit Feedback

- Unknown games (black-box games):
  - Each agent does not know the number, actions, and feedback of other agents
  - Each agent does not know the underlying game structure
  - Each agent can only observe the feedback of **the played action (bandit feedback)**, which is a joint result of all agents' actions



Correlated Equilibrium

Reward Matrix

	Action 1	Action 2
Action1	(0,0)	(0, 0.8)
Action 2	(0.8,0)	(-0.2, -0.2)

Joint Distribution

	Action 1	Action 2
Action 1	0	1/2
Action 2	1/2	0

The joint distribution of all agents' actions is a correlated equilibrium if no one is willing to deviate

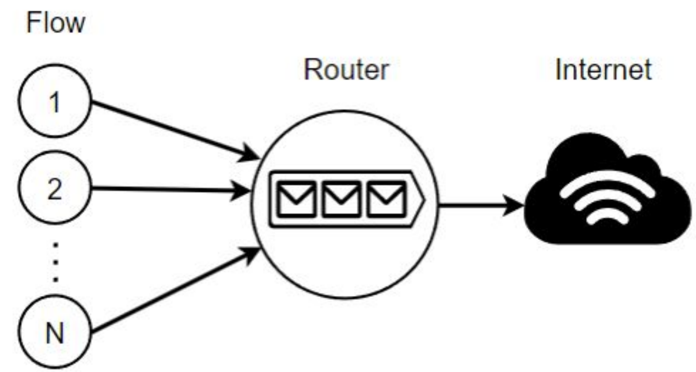
End-to-end Congestion Control as Unknown Games

Agent: Data flows

Actions: Congestion Window / Sending Rate

Utility: Reward functions considering

throughput and RTT



- The number of data flows may not be known a priori
- The actions of other data flows cannot be observed
- The only observed thing is the feedback such as packet loss and rtt, which can help calculate the utility

Objective 1

- Be as good as always playing the optimal action in hindsight by minimizing the "external regret":

$$R_n^{\text{ext}}(T) := \max_{w' \in W_n} \sum_{t=1}^{T} u_n\left(w'; \mathbb{W}_{-n}^t\right) - \sum_{t=1}^{T} \sum_{w \in W_n} \mathbf{1}\left[w_n^t = w\right] u_n\left(w; \mathbb{W}_{-n}^t\right)$$

W_n Actions set for agent *n*

w_n^t Action played by agent *n* in round t

$\mathbb{W}_{-n}^t$ Actions played by agents other than agent *n*

Objective 1

- Be as good as always playing the optimal action in hindsight by minimizing the “external regret”:

$$R_n^{\text{ext}}(T) := \max_{w' \in W_n} \sum_{t=1}^{T} u_n(w'; \mathbb{W}_{-n}^t) - \sum_{t=1}^{T} \sum_{w \in W_n} \mathbf{1}\left[w_n^t = w\right] u_n(w; \mathbb{W}_{-n}^t)$$

utility observed by always playing *w’*

W_n Actions set for agent *n*

w_n^t Action played by agent *n* in round t

$\mathbb{W}_{-n}^t$ Actions played by agents other than agent *n*

Objective 1

- Be as good as always playing the optimal action in hindsight by minimizing the “external regret”:

$$R_n^{\text{ext}}(T) := \max_{w' \in W_n} \sum_{t=1}^{T} u_n\left(w'; \mathbb{W}_{-n}^t\right) - \sum_{t=1}^{T} \sum_{w \in W_n} \mathbf{1}\left[w_n^t = w\right] u_n\left(w; \mathbb{W}_{-n}^t\right)$$

utility observed by always playing *w’*

utility observed by agent n playing a learning algorithm

W_n Actions set for agent *n*

w_n^t Action played by agent *n* in round t

$\mathbb{W}_{-n}^t$ Actions played by agents other than agent *n*

Objective 2

- Converge to the correlated equilibrium by minimizing the “internal regret”:

$$R_n^{\text{int}}(T) := \max_{w,w' \in W_n} \sum_{t=1}^{T} \mathbf{1}\left[w_n^t = w\right]\left(u_n\left(w'; \mathbb{W}_{-n}^t\right) - u_n\left(w; \mathbb{W}_{-n}^t\right)\right)$$

W_n Actions set for agent *n*

w_n^t Action played by agent *n* in round t

$\mathbb{W}_{-n}^t$ Actions played by agents other than agent *n*

Objective 2

- Converge to the correlated equilibrium by minimizing the “internal regret”:

$$R_n^{\text{int}}(T) := \max_{w,w' \in W_n} \sum_{t=1}^{T} \mathbf{1}\left[w_n^t = w\right] \left(u_n\left(w'; \mathbb{W}_{-n}^t\right) - u_n\left(w; \mathbb{W}_{-n}^t\right) \right)$$

The reward of playing *w’*

W_n Actions set for agent *n*

w_n^t Action played by agent *n* in round t

$\mathbb{W}_{-n}^t$ Actions played by agents other than agent *n*

Objective 2

- Converge to the correlated equilibrium by minimizing the "internal regret":

$$R_n^{\text{int}}(T) := \max_{w,w' \in W_n} \sum_{t=1}^{T} \mathbf{1}\left[w_n^t = w\right]\left(u_n\left(w'; \mathbb{W}_{-n}^t\right) - u_n\left(w; \mathbb{W}_{-n}^t\right)\right)$$

The reward of playing *w'* The reward of playing *w*

W_n Actions set for agent *n*

w_n^t Action played by agent *n* in round t

$\mathbb{W}_{-n}^t$ Actions played by agents other than agent *n*

Swap Regret

Minimize the external and internal regret simultaneously:

$$R_n^{\mathrm{swa}}(T, \mathcal{F}_n) = \max_{F \in \mathcal{F}_n} \sum_{t=1}^{T} \sum_{w \in W_n} \mathbf{1}\left[w_n^t = w\right] u_n\left(F(w); \mathbb{W}_{-n}^t\right) - \sum_{t=1}^{T} \sum_{w \in W_n} \mathbf{1}\left[w_n^t = w\right] u_n\left(w; \mathbb{W}_{-n}^t\right)$$

W_n	Actions set for agent *n*	$\mathbb{W}_{-n}^t$	Actions played by agents other than agent *n*
w_n^t	Action played agent *n* in round t		

Swap Regret

Minimize the external and internal regret simultaneously:

$$R_n^{\mathrm{swa}}(T, \mathcal{F}_n) = \max_{F \in \mathcal{F}_n} \sum_{t=1}^{T} \sum_{w \in W_n} \mathbf{1}\left[w_n^t = w\right] u_n\left(F(w); \mathbb{W}_{-n}^t\right) - \sum_{t=1}^{T} \sum_{w \in W_n} \mathbf{1}\left[w_n^t = w\right] u_n\left(w; \mathbb{W}_{-n}^t\right)$$

Rewards observed by the learning algorithm

W_n Actions set for agent *n*

w_n^t Action played by agent *n* in round t

$\mathbb{W}_{-n}^t$ Actions played by agents other than agent *n*

Swap Regret

Minimize the external and internal regret simultaneously:

$$R_n^{\text{swa}}(T, \mathcal{F}_n) = \max_{F \in \mathcal{F}_n} \sum_{t=1}^{T} \sum_{w \in W_n} \mathbf{1}\left[w_n^t = w\right] u_n\left(F(w); \mathbb{W}_{-n}^t\right) - \sum_{t=1}^{T} \sum_{w \in W_n} \mathbf{1}\left[w_n^t = w\right] u_n\left(w; \mathbb{W}_{-n}^t\right)$$

Rewards observed by a competitor plays an action F(w)

Rewards observed by the learning algorithm

W_n Actions set for agent *n*

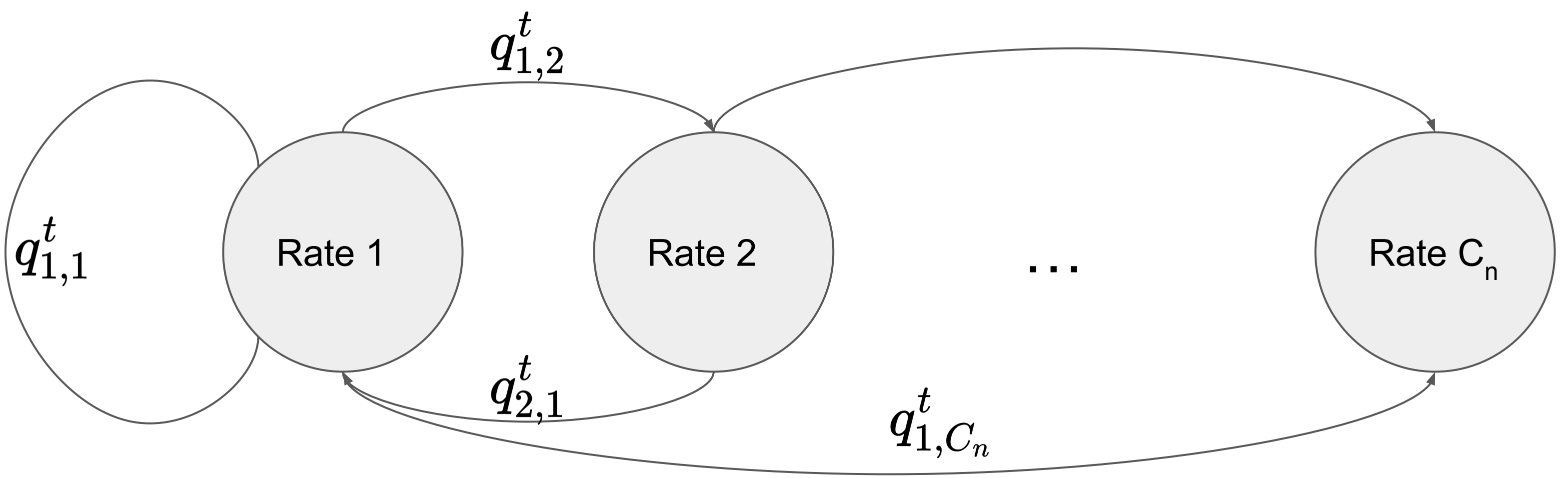
w_n^t Action played by agent *n* in round t

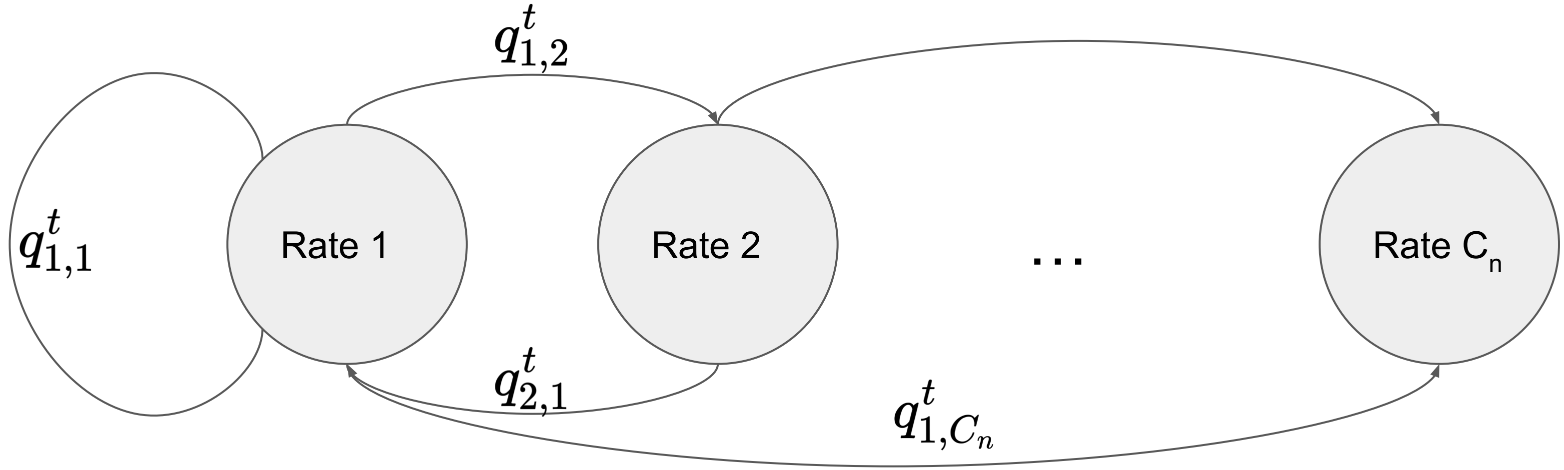
$\mathbb{W}_{-n}^t$ Actions played by agents other than agent *n*

Main Idea of The LUC algorithm

p_w^t The probability of choosing action (cwnd/rate) *w*

$q_{w,w'}^t$ The probability of choosing w’ instead of choosing w





$$\begin{bmatrix} p^t_1 & p^t_2 & \cdots & p^t_{c_n} \end{bmatrix} = \begin{bmatrix} p^t_1 & p^t_2 & \cdots & p^t_{c_n} \end{bmatrix} \begin{bmatrix} q^t_{1,1} & q^t_{1,2} & \cdots & q^t_{1,C_n} \\ q^t_{2,1} & q^t_{2,2} & \cdots & q^t_{2,C_n} \\ \vdots & & \ddots & \\ q^t_{C_n,1} & q^t_{C_n,2} & \cdots & q^t_{C_n,C_n} \end{bmatrix}$$

By calculating the stationary distribution, we obtain the selection distribution

Analytical Results

Theorem 1: Swap regret is bounded by

$$O\left(C_n\sqrt{T\log(C_n/\delta)}\right)$$

with probability at least 1- δ

C_n The number of actions

Theorem 2: If every flow plays LUC for *T* rounds, the empirical distribution of the joint actions played by all flows

$$\hat{\mathbf{P}}^T(\mathbb{W}) := \frac{1}{T}\sum_{t=1}^{T}\mathbf{1}[\mathbb{W}_t = \mathbb{W}]$$

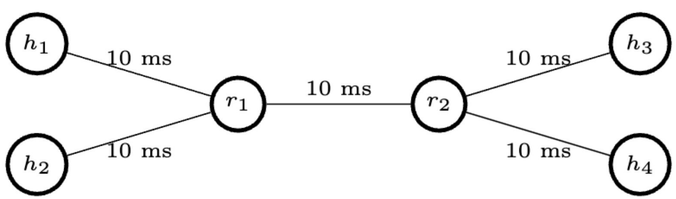
is an *ε*-correlated equilibrium with probability at least 1- *δ*

Emulation Results

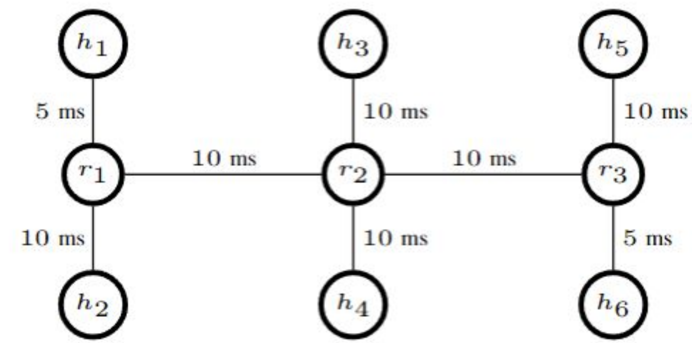
We have implemented LUC in Linux Kernel 5.4.0 based on the congestion control plane, a new API for writing congestion control algorithms.

Compare with CUBIC, BBR2

Emulation on Mininet, link capacity 50mbps

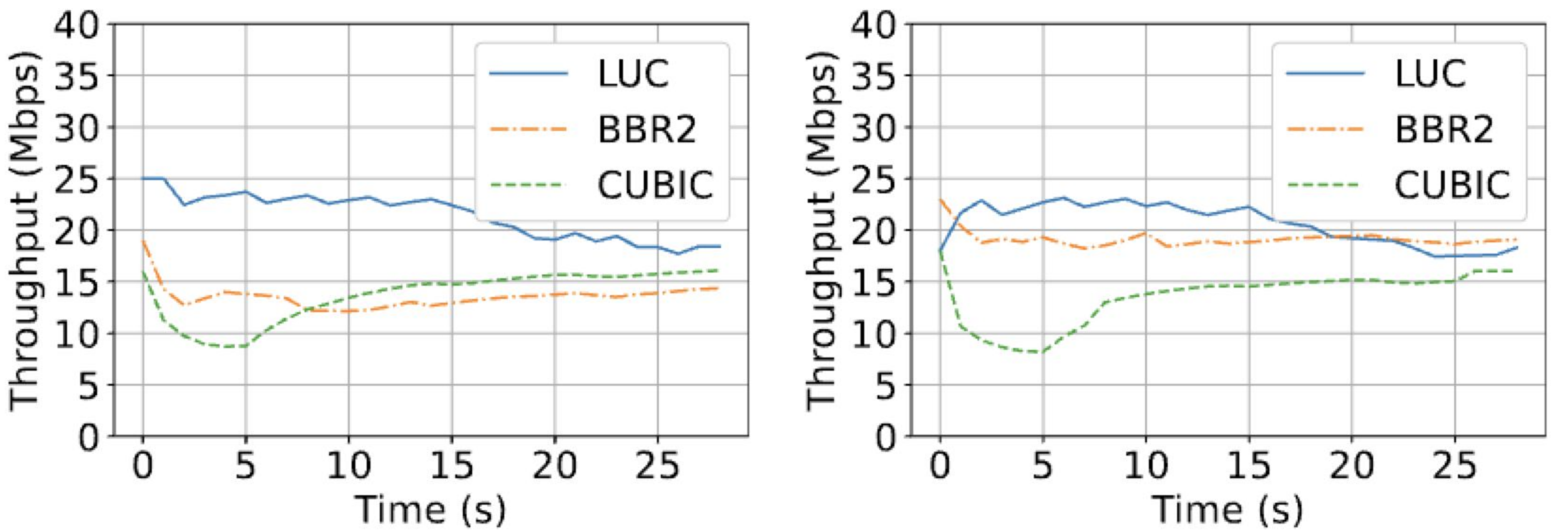


(a) The dumbbell topology



(b) The parking lot topology

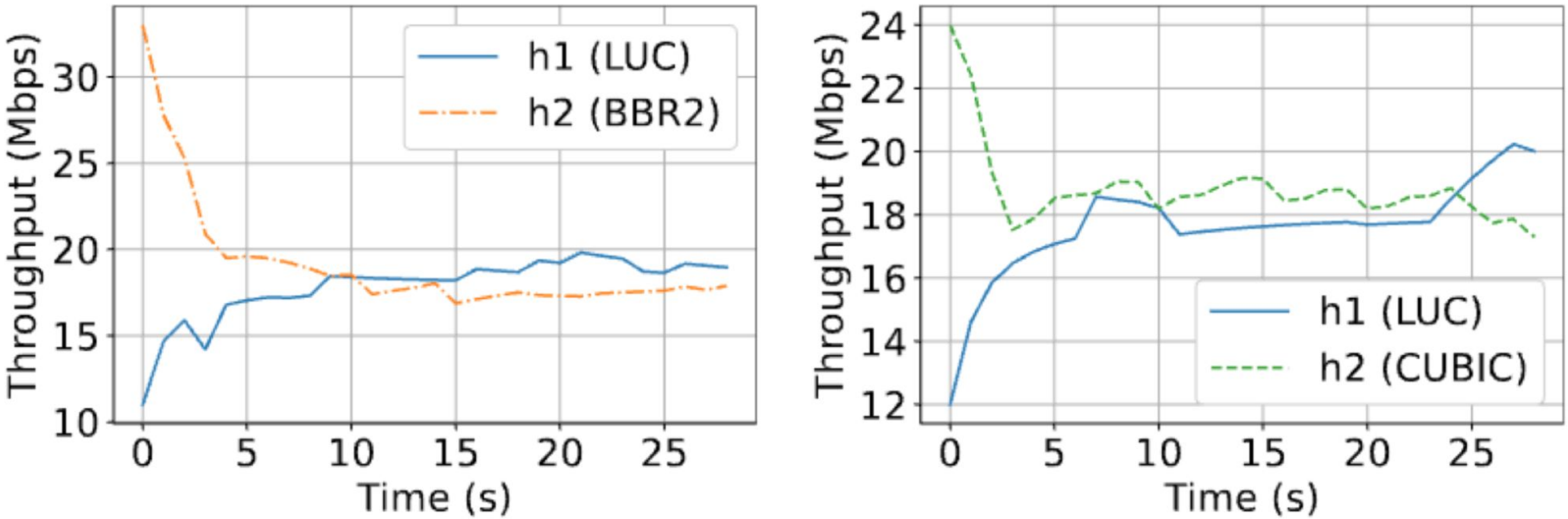
Dumbbell Result



(a) Homo-flow Throughput on h_1

(b) Homo-flow Throughput on h_2

- When both flows adopt LUC, they can achieve the similar performance



(c) Throughput (LUC, BBR2) (d) Throughput (LUC, CUBIC)

- LUC is competitive and TCP-friendly

Limitations and Future Works

- Relax the assumptions that all flows can finish a transmission in each round
- Address the large action set in real-world communications
- Apply the swap-regret-minimizing technique as a building block to improve other algorithms such as BBR

Thanks

Backup Slides

Single-agent MABs vs Multi-agent MABs

Single-agent MAB:

- The reward/loss of actions in round *t* is determined **at the beginning of round *t***

Multi-agent MAB:

- The reward/loss of actions in round *t* is determined **by the end of round *t* due to the dependence on all agent's actions**

LUC

Algorithm 1 The LUC algorithm

procedure LUC$(n, W_n, \eta, \beta, \lambda, P_0)$

// Initialization

Set $q^1_{w,w'} = \frac{1}{C_n}$ and $\hat{S}^0_{w,w'} = 0, \forall w, w' \in W_n$

for $t = 1, 2, 3, \ldots$ **do**

Calculate the distribution on the action set by solving the equation $P^t_n = P^t_n \mathbb{Q}^t_n$

Choose a $w^t_n \sim P^t_n$ and send packets accordingly

Observe feedback and calculate utility u^t_n for the chosen w^t_n

// Update each meta-distribution

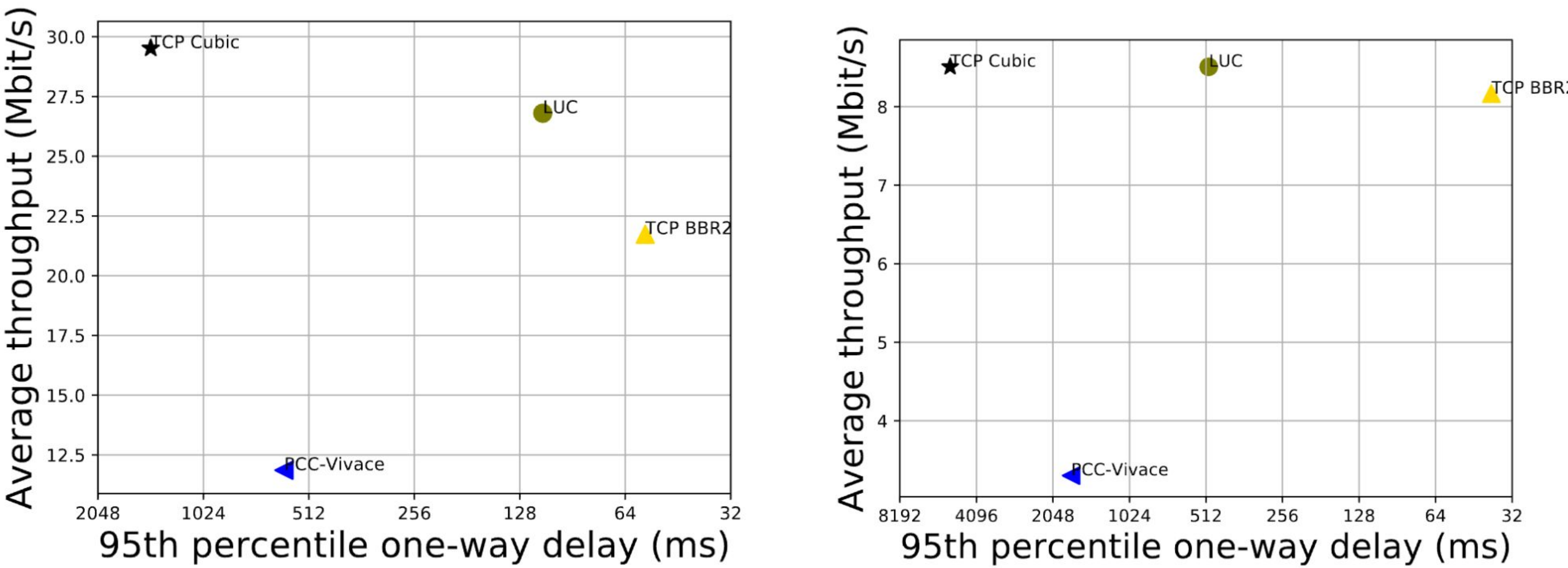
for $w \in W_n$ **do**

Calculate $\hat{X}^t_{w,w'}, \forall w' \in W_n$ based on $\hat{X}^t_{w,w'} := \frac{\mathbf{1}[w^t_n = w'] p^t_w \left(x^t_{w'} + \beta\right)}{p^t_{w'}}$

$\hat{S}^t_{w,w'} = \hat{S}^{t-1}_{w,w'} + \hat{X}^t_{w,w'}, \forall w' \in W_n$

Calculate Q^{t+1}_w based on

$$q^{t+1}_{w,w'} = (1-\lambda) \frac{\exp\left(\eta \hat{S}^t_{w,w'}\right)}{\sum\limits_{w'' \in W_n} \exp\left(\eta \hat{S}^t_{w,w''}\right)} + \lambda P_0$$



(a) T-mobile LTE Network

(b) Verizon LTE Network

Fig. 6: The trace-driven experiment results on Pantheon.

Time and Space Complexity

For each agent n, the time complexity is dependent only on its own action set C_n

Time complexity: $O(C_n^2)$

Space complexity: $O(C_n^2)$