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

Zhiming Huang
Department of Computer Science
University of Victoria, Canada
Email: zhiminghuang@uvic.ca

Kaiyang Liu*
Department of Computer Science
Memorial University of Newfoundland, Canada
Email: liukaiyang@uvic.ca

Jianping Pan
Department of Computer Science
University of Victoria, Canada
Email: pan@uvic.ca

***Abstract*—In this paper, we study the open problems raised by Karp et al. in FOCS 2000, where the authors formulated the end-to-end congestion control as a repeated game between a flow and an adversary. They mentioned several open problems including finding equilibria in a more realistic game model for the situation where the available bandwidth is a result of competition among multiple flows instead of being chosen by an adversary, and designing the randomized algorithm to deal with the dynamic change of network bandwidth. Although there have been many game-theoretic works for congestion control, to the best of our knowledge, the above two problems still remain unsolved over the past decades. We take a step further to address the above two problems by first modeling the end-to-end congestion control as a repeated unknown general-sum game among multiple flows with bandit feedback. Each flow is a player in this unknown game, making decisions on how many packets to send. The throughput for each flow depends on all the flows' rates and the network capacity. The unknown setting and bandit feedback capture the essence of end-to-end congestion control: each flow has no information about others (e.g., the number, actions, and packet loss of other flows), and only receives limited information for its chosen action. Then, we propose a randomized no-regret learning algorithm for each flow called LUC based on a swap-regret-minimizing technique. We prove that LUC can guarantee a polynomial-time convergence rate to correlated equilibria in the multi-player setting. Finally, we have implemented LUC through the Linux kernel, and conducted extensive fairness-related experiments in Mininet and trace-driven experiments with Pantheon to show that each flow with LUC can fairly share the bandwidth in homogeneous scenarios, and be competitive but TCP-friendly in heterogeneous scenarios.**

I. Introduction

The study of congestion control in computer networks has kept prosperous due to its intrinsic complexity in the distributed control of flows with limited information. The modern Internet design philosophy of end-to-end delivery results in the decision making at the end-host, forming a strategic environment for resource competition in networks. Studying such a strategic environment is the core of the algorithmic game theory, and thus it plays an important role in understanding the nature of the network congestion and analyzing the performance of congestion control algorithms [1].

Although many game-theoretic works have been done for congestion control, most of them are focused on analyzing the existing TCP congestion control algorithms or router policies (e.g., drop-tail) [2]–[12]. Those works usually assume game models with all information available, e.g., the number of flows, the strategies, and the router policies are known a priori. Such game models work well when designing router policies, as a router has information about incoming flows. However, when it comes to the design of end-to-end congestion control algorithms, such game models fail to capture the reality where each flow has limited information about others and can only observe the outcome for its chosen *congestion window (cwnd)* or *sending rate (srate)*.

The very first game model for designing end-to-end congestion control algorithms is proposed by Karp et al. [13] in FOCS 2000 where the authors formulated the end-to-end congestion control as a repeated game between a flow and an adversary. In each round of the game, the flow sends a *cwnd* of packets to the network, and the adversary chooses available network bandwidth for the flow but the flow cannot observe it. Then by the end of the round, the flow will observe a utility determined by the number of sent packets and the available network bandwidth. Such a model is simple yet effective to capture the interaction between one flow and the network. However, available bandwidth is simply assumed to be dynamically chosen by an adversary, while in reality, the dynamic of available bandwidth is a result of competition among multiple flows. Thus, the author of [13] proposed several open problems including finding equilibria in a more realistic game model considering the competition among multiple flows, and designing randomized algorithms to deal with the dynamic available bandwidth.

Over the past decades, the above open problems still remain unsolved. Although there are many works in recent years that adopt online learning techniques to design end-to-end congestion control algorithms [14]–[19], they are either explicitly or implicitly based on the simple model proposed in [13]. The PCC-Vivace [20] algorithm, on the other hand, implicitly formulates the end-to-end congestion control as a concave game based on the theoretical results in [21]: when minimizing the so-called external regret in concave games for each player, all players will reach Nash equilibria. Albeit concave games capture some game-theoretic essence of the end-to-end congestion control, the assumption about the concave

*Portions of this work were completed while the author was a postdoctoral fellow at the University of Victoria, BC, Canada.

utility function is quite strong. In addition, *external regret* is a performance metric measuring the maximum performance loss between an online learning algorithm and a set of competitors always playing a fixed action, but even the best fixed action may not be the optimal solution for the game. Thus, a more realistic game model and learning algorithms are needed to address the open problems.

We take a step further for the open problems by formulating end-to-end congestion control as learning for repeated unknown general-sum games with bandit feedback. In each round of the unknown general-sum games, or equivalently, black-box games [22], each flow needs to make decisions *independently* in a *distributed* manner with *limited information*. The limited information means that each flow may not know the number of all the flows in the same network, and each flow cannot observe the information (e.g., the congestion window and packet loss) about other flows, or communicate with other flows. The bandit feedback means that each flow can only know its own utility (e.g., throughput) for its chosen *cwnd* or *srate*. The objective of each flow is to accumulate as many utilities as possible, and all the flows converge to equilibria. Instead of Nash equilibria, we consider a generalization of Nash equilibrium called the correlated equilibrium [23]. The correlated equilibria usually require a central controller to give recommendations to the agents involved in the game so that the system can achieve maximum efficiency. However, in the unknown games, we do not assume that each flow can obtain any recommendations. Is there any strategy if played by all the flows can achieve the correlated equilibria as if there were a central controller?

This problem is very challenging as flows are affecting each other, and each flow with such limited information needs to trade off between exploring (i.e., probing) the reward for each *cwnd* (or *srate*) and exploiting the current knowledge learned from the exploration to make the best decisions. Motivated by the swap regret [24], [25], a generic performance measure for online learning algorithms, we develop a randomized algorithm called *Learning for Unknown games for Congestion Control (LUC)* for each flow in the unknown games with bandit feedback. We show that LUC is a no-swap-regret learning algorithm, i.e., the time-averaged swap regret vanishes asymptotically over time. The advantages of minimizing swap regret are twofold. First, a no-swap-regret learning algorithm is robust to a larger set of competitors (see more discussions in Sec. III). Second, minimizing swap regret is a computationally-efficient way to find a correlated equilibrium (see Corollary 1). LUC is designed to be a building block that can be used to design end-to-end congestion control algorithms that address the unknown games and achieve correlated equilibria efficiently.

To sum up, the contributions of our work are as follows:

- We are the first in the literature to formulate the end-to-end congestion control as repeated unknown general-sum games with bandit feedback, which take a step further to address the open problems raised in [13] by capturing more game-theoretic essence of the end-to-end congestion control in reality.
- We proposed a polynomial-time randomized algorithm called LUC to address the unknown games, which has a cumulative swap regret upper bounded by $O(C_n\sqrt{T\log(C_n\delta^{-1})})$ with probability at least $1-\delta$ for any $\delta \in (0,1)$, where C_n is the number of actions for flow n and T is the total length of the game. Furthermore, the LUC algorithm can achieve an ϵ-correlated equilibrium in a polynomial number of rounds.
- Third, we implement LUC through the Linux kernel 5.4.0 based on the congestion control plane [26], a new API for writing congestion control algorithms. We first perform TCP fairness-related experiments in Mininet to compare with the TCP CUBIC and TCP BBR version 2. Then we perform experiments driven by U.S. cellular network traces with Pantheon [27] with an additional comparison to PCC-Vivace [20]. The experiment results show that LUC is TCP-friendly and competitive, and can adapt to dynamic network environments.

The rest of the paper is organized as follows. Sec. II reviews related works. The problem settings are described in Sec. III. The LUC algorithm is proposed in Sec. IV, with analytical results presented in Sec. V. We show the throughput and fairness-related experiments in Sec. VI. Sec. VII concludes the paper. The detailed proofs of the swap-regret upper bound are deferred to Appendix.

II. Related Works

In this section, we first give a detailed review of the game-theoretic congestion control, and then briefly discuss recent progresses in learning-based congestion control. Furthermore, we will review equilibrium learning in game theory.

Game-theoretic Congestion Control: Game theory has been extensively studied in congestion control, and there are mainly two lines of research for game-theoretic congestion control. One is focused on router-based congestion control, which manages the incoming packets from different flows for a router, and the other studies the end-to-end congestion control, which decides how many packets to be sent at a time for each flow.

For router-based congestion control, the main goal is to analyze the existing TCP congestion control algorithms with given router policies (e.g., drop-tail) or design new router policies. The earliest work can be traced back to [28], which gave game-theoretic implications of switching disciplines and their relevance to congestion control. It was later followed by the works of [2]–[10], where the independent data flows are considered as selfish players in a game, and the mechanism of the game is determined by the router policies. In such games, both the router policies and the end-to-end congestion control algorithms (i.e., the strategies of players) are known a priori. Although the above works can be effective to design and analyze router policies, they cannot provide an end-to-end congestion control solution for data flows.

The other line of research studies the design and analysis of end-to-end congestion control algorithms from a game-theoretic point of view. One of the earliest works is [13], where the author modeled the congestion control problem as a game between a flow and an adversary. In each round, a flow selects an action (e.g., *cwnd*) and the adversary selects a bandwidth for triggering penalties if congestion happens or there is wasted bandwidth. By the end of that paper, they raised several open questions including how to model a more realistic case where the available bandwidth is a result of the competition among flows instead of being chosen by the adversary, and whether equilibria exist in such scenarios. Such a multi-flow game problem is challenging, as each flow has very limited information about other flows in the end-to-end congestion control.

Later, in the work of [29], the author modeled the end-to-end congestion control as a noncooperative game, and proved the existence and uniqueness of Nash equilibrium with convexity assumptions for utility functions. They also design gradient algorithms to achieve the Nash equilibrium. However, the gradient algorithm requires knowledge about the total number of flows and the network capacity, which is not practical in reality. To address this issue, the authors of [30] tried to model the end-to-end congestion control as a Bayesian game. Although each flow does not need to know the exact information about other flows, a prior belief about others is still required.

Some theoretical progress has been made in the work of [20], where the authors designed PCC-Vivace based on the theoretical work [21] for equilibrium learning in concave games, i.e., the utility function for each flow is concave, where the Nash equilibrium can be reached if each agent plays an external-regret-minimizing algorithm. However, it is not realistic to assume the utility for each flow must be a concave function. On the other hand, unknown general-sum games can well capture the game-theoretic nature of end-to-end congestion control, as there are no unrealistic assumptions for either the flows or the utility functions.

To the best of our knowledge, the open problem proposed by [13] has still remained unsolved. We take a step further by modeling the competition of multiple flows as a repeated unknown general-sum game with bandit feedback for the first time in literature and proposing a swap-regret-minimizing algorithm to asymptotically achieve the correlated equilibria that are more general than the well-known Nash equilibria.

Learning-based Congestion Control: Recent years have witnessed a line of research work on congestion control based on machine learning techniques. Remy [31] and Indigo [27] are two representative works for offline-learning congestion control algorithms. Such algorithms have limited adaptivity to new situations in practice. Therefore, *reinforcement learning (RL)* techniques have been introduced to congestion control to alleviate such problems, such as QTCP [14], Aurora [15], Eagle [16], Orca [17], MOCC [18] and Pareto [19]. However, a certain amount of offline training is often needed for the above RL models to guarantee an efficient and effective deployment. Lightweight online learning techniques do not require a pre-trained model. The typical example is PCC-Vivace [20], which relies on online (convex) optimization to update the sending rate. Although the above works share some similarities to the equilibrium learning in our work, the common limitation of the above learning-based algorithms is that they can only minimize external regret, i.e., the maximum performance gap from the set of competitors always playing a fixed action is bounded. There are no theoretical guarantees for the convergence to correlated equilibria.

Equilibrium Learning: The study of unknown game models (or the black-box games [22]) has a long history that can be traced back to the fictitious play for the two-player zero-sum games [32], [33]. However, it was not until the start of this century that much progress has been made with the development of online learning techniques [34], particularly for games with specific structures. For example, the authors of [35] studied the congestion game with bandit feedback, where the author tried to minimize a Nash regret, which is a sum of the maximal external regret among agents in each round. A similar work of [36] studied a specific congestion game, where each resource is equally shared among the agents who choose it. Nevertheless, end-to-end congestion control is not necessarily a congestion game, as one may not find a potential function that is the essence of the congestion game. The authors of [37] studied augmented games by utilizing communications between agents. However, such a methodology is not suitable for unknown games, as the agents in unknown games do not know each other and will not communicate with each other. There are many other equilibrium learning works for some specific games, e.g., potential games [38]–[41], and mean-field games [42]–[44]. As all their results require specific game structures, they cannot be applied to unknown general-sum games.

Regarding the learning for unknown general-sum games, there are mainly two situations depending on the observability of feedback. If the utility of an action can be observed regardless of whether it is played or not, we call it the *full-information feedback* [45]–[47], and if only the utility of a played action can be observed, then it is the *bandit feedback*. As in end-to-end congestion control, each flow can only observe the feedback for its selected *cwnd* (or *srate*), we will focus on learning for the bandit feedback.

The first work that addressed the unknown general-sum game problem with bandit feedback is [48], where an exponential-weight technique is proposed to minimize external regret. However, it is shown in [34] that the external-regret-minimizing algorithm can only converge to the set of Nash equilibria for the two-person zero-sum game. It was later proved in [21] that minimizing external regret can converge to Nash equilibria for concave games. However, for end-to-end congestion control, we cannot take for granted that the utility function is concave. As we want to come up with a building block that can be adapted to any congestion control algorithm, we are more interested in unknown general-sum games, and the correlated equilibrium can only be achieved if the internal regret can be minimized asymptotically. Since minimizing the swap regret can also minimize both the external and internal regret [24] and be more robust against a larger set of competitors, we are motivated to address the unknown games for end-to-end congestion control from the swap-regret viewpoint, which is different from the Nash regret that is prevalent in the equilibrium-learning literature.

III. Model and Problem Formulation

A. Unknown General-sum Game Model with Bandit Feedback

We consider a network of N flows competing for the same resource (e.g., bandwidth) in the network, as shown in Fig. 1. The congestion happens when the number of packets sent by all flows is beyond the network capacity, and the overflowed packets will be dropped according to a router policy (e.g., drop-tail). The competitive interaction between multiple flows can be modeled by a repeated unknown general-sum game involving N flows with the following two justifications that are also widely used in the related works [3], [10].

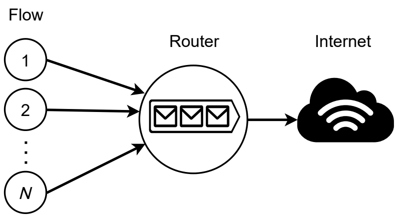


Fig. 1: A network model for the unknown general-sum games.

First, the length of each round of the repeated game is chosen appropriately such that all the flows can finish one round of interaction in one round of the game, i.e., send a *cwnd* of packets and receive ACKs for these packets. Second, we assume the packet loss is only caused by the congestion, as the congestion control scheme in the current TCP also assumes the packet loss is congestion-induced.

Therefore, we can formally define the repeated unknown general-sum game with bandit feedback for end-to-end congestion control as follows. Denote by $\mathcal{N} := \{1, \dots, N\}$ the set of all flows in the game, and each flow $n \in \mathcal{N}$ is associated with a (possibly different) finite set of actions (i.e., *cwnd* or *srate*) $W_n := \{C_{\min}, \dots, C_{\max}\}$, where $C_{\max}$ is the maximum *cwnd* (or *srate*) of flow n. Let $C_n := |W_n|$ be the total number of actions for flow n. In an actual implementation, $C_{\min}$ and $C_{\max}$ can be either determined by an initial probing phase, or determined by the network physical-layer capabilities. The game is repeated for T rounds. In each round $t = 1, \dots, T$, each flow selects a *cwnd* (or *srate*) $w_n^t \in W_n$. By the end of round t, each flow observes feedback information (e.g., packet loss and round trip time) and calculates utility $u_n^t \in [0, 1]$. We only require that the calculated utility should be normalized between 0 and 1, and do not give any specific form of the utility function here because we want our solution to be general enough for any utility functions, so that the solution to the unknown games can be acting as a building block for other congestion control algorithms.

For any given network capacity and router policies, the utility for each flow n is not only dependent on its own action w_n^t but also determined by the actions of all other flows. Denote by $\mathbb{W}^t := \{w_n^t : \forall n \in \mathcal{N}\}$ the *action profile* in round t. To emphasize the dependency of the utility on all the flows, we further write u_n^t as $u_n(\mathbb{W}^t)$ or $u_n(w_n^t, \mathbb{W}_{-n}^t)$, where $(w_n^t; \mathbb{W}_{-n}^t)$ is an abbreviation of $\mathbb{W}^t := (w_1^t, \dots, w_n^t, \dots, w_N^t)$ with a highlight of flow n's action w_n^t against other flows' actions.

We do not directly model the flow importance (e.g., the QoS requirements) in the game, as the importance of the flows is taken into consideration by router policies. For example, one can design a router policy that drops fewer packets for the flows with a higher QoS requirement. As the focus of this paper is the design of a solution for the unknown game with any router policies, our solution still works even if the flows are of different importance.

Note that each flow is in a *bandit feedback setting*, i.e., neither the actions nor the loss of other flows can be observed, and each flow n can only observe the information such as packet loss and round trip time to calculate its own utility for the chosen *cwnd* (or *srate*). Also, neither the number of flows nor the router policy is known a priori to each flow. The reason for considering such a limited information setting is to make the model more realistic so that our algorithm is more deployable to the end systems without modifying the intermediate nodes.

B. Problem Formulation

The goal of each flow in the unknown general-sum games with bandit feedback is to accumulate as many utilities as possible without getting the network congested. Network congestion results in packet loss and queuing delay, which further reduces the utilities for each flow. Such a goal can be easily achieved if a router can act as a central controller to allocate *cwnd* (or *srate*) for each flow by sending control messages. For example, if all the N flows are of the same importance, i.e., all the N flows have the same QoS requirements, then the optimal *srate* for each flow is C/N, where C is the network capacity. To accommodate more general situations where the importance of flows can be different or time-varied, we use the notion of *ϵ-correlated equilibrium* to measure the optimality of a solution, as defined below.

Definition 1. *Let $\mathbf{P}$ be a joint probability distribution over $\mathcal{W}$, where $\mathcal{W} := \prod_{n \in \mathcal{N}} W_n$ is the space spanned by all combinations of flows' actions. We say $\mathbf{P}$ is an ϵ-correlated equilibrium if the expected incentive for each flow n to deviate from action w to another action w' is no more than $\epsilon \geq 0$, i.e., $\forall n \in \mathcal{N}$, we have*

$$\sum_{(w; \mathbb{W}_{-n}) \in \mathcal{W}} \mathbf{P}(\mathbb{W}) \left(u_n(w'; \mathbb{W}_{-n}) - u_n(\mathbb{W}) \right) \leq \epsilon. \quad (1)$$

Intuitively, the router draws an action profile from $\mathbf{P}$ and privately recommends the *srate* to each flow. For example, in the case of flows with equal importance, $\mathbf{P}(w_n = \frac{C}{N}, \forall n \in \mathcal{N}) = 1$ and the probabilities for drawing other profiles are equal to 0. As no flow will gain more than ϵ to choose a different *srate*, provided that other flows follow the router's recommendation, such a $\mathbf{P}$ is an ϵ-correlated equilibrium. Compared with other forms of equilibria, correlated equilibrium considers the joint instead of the marginal distribution of the action space and does not assume independence among different action sets. Thus, correlated equilibrium is more general and useful.

However, the goal of each flow to accumulate maximum utilities becomes more challenging when each flow in the unknown games makes decisions independently. The problem of learning for unknown general-sum games is stated as follows: When the only information revealed to each flow is the utility of its chosen *cwnd* (or *srate*) in each round, is there any algorithm that can help each flow accumulate more utilities and converge to the ϵ-correlated equilibrium as if there were a central controller?

More specifically, the ability to accumulate more utilities is measured by the *external regret*, which is defined to be the maximum performance loss between a learning algorithm and a set of competitors always playing a fixed action throughout the game. Let $\mathbf{1}[w_n^t = w]$ be the indicator function that returns 1 if w is the chosen *cwnd* (or *srate*) in round t by flow n and 0 otherwise. The external regret $R_n^{\text{ext}}(T)$ for flow n by the end of round T is defined as

$$R_n^{\text{ext}}(T) := \max_{w' \in W_n} \sum_{t=1}^{T} u_n(w'; \mathbb{W}_{-n}^t) - \sum_{t=1}^{T} \sum_{w \in W_n} \mathbf{1}[w_n^t = w] u_n(w; \mathbb{W}_{-n}^t), \tag{2}$$

On the other hand, we need to make sure that if all flows play the algorithm, the empirical distribution of the joint actions, denote by $\hat{\mathbf{P}}^T(\mathbb{W}) := \frac{1}{T} \sum_{t=1}^{T} \mathbf{1}\left[\mathbb{W}_t = \mathbb{W}\right], \mathbb{W} \in \mathcal{W}$, is an ϵ-correlated equilibrium defined in (1), where $\mathbf{P}^t$ is the joint distribution of actions in round t. Such an ability is measured by the *internal regret* [34], [49], which compares the actions of a flow in a pair-wise manner:

$$R_n^{\text{int}}(T) := \max_{w, w' \in W_n} \sum_{t=1}^{T} r_{(w,w'),n}^t, \tag{3}$$

where

$$r_{(w,w'),n}^t := \mathbf{1}[w_n^t = w] \left(u_n(w'; \mathbb{W}_{-n}^t) - u_n(w; \mathbb{W}_{-n}^t) \right)$$

is the instantaneous regret for flow n of having played arm w instead of arm w' in round t. In Theorem 2, we show that if all flows play an internal-regret-minimizing algorithm together, the empirical distribution of their joint actions is an ϵ-correlated equilibrium.

To minimize both external regret and internal regret at the same time, we introduce a more general notion of regret, called the *swap regret* [24]. By defining a swap function F_n for each agent n that maps one *cwnd* (or *srate*) $w \in W_n$ to another *cwnd* (or *srate*) $w' \in W_n$ and denoting by $\mathcal{F}_n$ a finite set of F_n, the swap regret for flow n with $\mathcal{F}_n$ up to round T is defined as follows:

$$R_n^{\text{swa}}(T, \mathcal{F}_n) = \max_{F \in \mathcal{F}_n} \sum_{t=1}^{T} \sum_{w \in W_n} \mathbf{1}[w_n^t = w] u_n(F(w); \mathbb{W}_{-n}^t) - \sum_{t=1}^{T} \sum_{w \in W_n} \mathbf{1}[w_n^t = w] u_n(w; \mathbb{W}_{-n}^t). \tag{4}$$

We can boil down the swap regret to the external regret by letting $\mathcal{F}_n := \{F_w : \forall w \in W_n\}$, where $F_w : W_n \to w$. The internal regret can be obtained by letting $\mathcal{F}_n := \{F_{(w,w')} : \forall w, w' \in W_n\}$, where $F_{(w,w')}(w) = w'$ and $F_{(w,w')}(w'') = w''$ for any other $w'' \in W_n$. Thus, a no-swap-regret learning algorithm has a bounded performance gap from $(C_n)^{C_n}$ competitors because there can be up to $(C_n)^{C_n}$ possible mappings in $\mathcal{F}_n$, while there are only C_n and C_n^2 mappings for external and internal regrets, respectively.

Therefore, we want to design an algorithm that can minimize the swap regret. Key notations for the whole paper are summarized in Table I.

TABLE I: Summary of Key Notations

Notations	Definition
N; $\mathcal{N}$	The number of flows; the set of all the flows
W_n; $\mathcal{W}$	The action set for flow n; the space of all combinations of flows' actions
C_n	The number of actions for flow n
$\mathbb{W}; \mathbb{W}^t$	An action profile; an action profile in round t
$u_n(\mathbb{W}^t)$; $u_n(\mathbb{W})$	The utility for flow n in round t given action profile $\mathbb{W}^t$; the expected utility for flow n conditioned on the action profile $\mathbb{W}$
t; T	The round of time; the total number of rounds
$P_n^t := \{p_w^t : \forall w \in W_n\}$	The probability distribution over W_n for mixed strategies
w_n^t	The *cwnd* selected by flow n in round t
$R_n^{\text{ext}}(T)$; $R_n^{\text{int}}(T)$; $R_n^{\text{swa}}(T)$	The external, internal and swap regret for flow n up to round T
$Q_{n,w}^t$	A meta-distribution in round t, i.e., $Q_{n,w}^t := \left[q_{w,w'}^t\right]_{w' \in W_n}$
$\mathbb{Q}_n^t$	The meta-distribution matrix with each row being a meta-distribution $Q_{n,w}^t$
$\hat{\mathbf{P}}^T$	The empirical distribution of joint actions of all flows over T rounds

IV. The LUC Algorithm

To address the problem of learning for the repeated unknown general-sum game with bandit feedback for end-to-end congestion control, one must balance the tradeoff between *exploration* and *exploitation*. The exploration means trying different actions to know more about the payoff for each action, and the exploitation means playing more often with the actions that might be optimal. Such a tradeoff has been thoroughly studied in the domain of multi-armed bandits. In our problem, each agent (i.e., flow) can be regarded as playing a non-stochastic multi-armed bandit against a non-oblivious adversary [48], as the utility for an agent is determined by all agents (flows). To tackle the non-oblivious adversary, one must randomize his/her decisions.

Therefore, the LUC algorithm uses a mixed strategy to choose actions, i.e., each $w \in W_n$ is chosen with probability p_w^t. Then, denote by $P_n^t := \left[p_w^t\right]_{w \in W_n}$ the distribution on the action set W_n, which is a row vector of the chosen probabilities and $\sum\limits_{w \in W_n} p_w^t = 1$. The basic idea of LUC is to assign more probabilities to the *cwnd* (or *srate*) with more utilities.

However, the solution in [48] can only minimize the external regret, which is not enough for our objectives including the convergence to correlated equilibria, as discussed in Sec. II. One must minimize the internal regret [34] to achieve the correlated equilibria.

To achieve our objectives, we adopt the swap-regret-minimizing framework by [24] to minimize the swap regret, which is the key to minimizing the external regret and internal regret at the same time. We first need to model all the actions in W_n to form a Markov chain with a transition matrix, denoted by $\mathbb{Q}_n^t$, being updated by the end of each round with the obtained utility. Then, the LUC algorithm chooses each $w \in W_n$ according to the stationary distribution of the Markov chain.

We create a transition probability vector called meta-distribution for each $w \in W_n$, denoted by $Q_{n,w}^t := \left[q_{w,w'}^t\right]_{w' \in W_n}$, where $\sum\limits_{w' \in W_n} q_{w,w'}^t = 1$. The transition matrix $\mathbb{Q}_n^t$ is a $C_n \times C_n$ matrix with each row being $Q_{n,w}^t$. Then, P_n^t is a solution to the following system of linear equations:

$$P_n^t = P_n^t \mathbb{Q}_n^t. \tag{5}$$

Intuitively, such a Markov chain models the situation where at the beginning of a round, agent n first chooses w', but before sending packets, the agent regrets choosing w' and chooses w instead ($w', w \in W_n$ and they can be identical). In such a way, the probability of directly choosing $w \sim P_n^t$ is equivalent to the probability of first choosing meta-distribution $Q_{n,w'}^t$ for any $w' \sim P_n^t$ and then choosing $w \sim Q_{n,w'}^t$.

The update for the transition probability matrix is done by the exponential-weight technique and the process is described as follows. After obtaining utility $x_{w'}^t := u_n^t(w'; \mathbb{W}_{-n}^t)$ for the chosen w_n^t, the reward of choosing w' instead of w is defined as follows:

$$\hat{X}_{w,w'}^t := \frac{\mathbf{1}[w_n^t = w'] p_w^t \left(x_{w'}^t + \beta\right)}{p_{w'}^t}, \tag{6}$$

which is a division of the total reward $x_{w'}^t$ plus a bias parameter $\beta \in [0, 1]$ according to the stationary distribution.

Denote by $\hat{S}_{w,w'}^t = \hat{S}_{w,w'}^{t-1} + \hat{X}_{w,w'}^t$ the variable for tracking the biased estimated reward pair $w, w' \in W_n$. Then, with $\hat{S}_{w,w'}^t$, we can update the transition probability from w to w' as follows:

$$q_{w,w'}^{t+1} = (1 - \lambda) \frac{\exp\left(\eta \hat{S}_{w,w'}^t\right)}{\sum\limits_{w'' \in W_n} \exp\left(\eta \hat{S}_{w,w''}^t\right)} + \lambda P_0, \tag{7}$$

where η_t is a non-increasing and positive parameter controlling the learning rate, P_0 is the initial distribution among actions learned from past experience, and $\lambda \in (0, 1)$ is a tradeoff parameter. Thus, with P_0, we can utilize offline training with traces to improve learning efficiency. If learning from scratch, we can let P_0 be the uniform distribution among actions. The possible values of those parameters are given in Theorem 1. Note that $\hat{S}_{w,w'}^t$ will not increase if w' is not chosen in round t because of the definition of $\hat{X}_{w,w'}^t$. This shows that (7) helps trade off between exploration and exploitation: if an action is not chosen for a long time or the action always suffers a lower loss, the probability of choosing it will also increase.

Algorithm 1 The LUC algorithm

```
1: procedure LUC($n, W_n, \eta, \beta, \lambda, P_0$)
   // Initialization
2:     Set $q_{w,w'}^1 = \frac{1}{C_n}$ and $\hat{S}_{w,w'}^0 = 0, \forall w, w' \in W_n$
3:     for $t = 1, 2, 3, \ldots$ do
4:         Calculate the distribution on the action set by solving the equation $P_n^t = P_n^t \mathbb{Q}_n^t$
5:         Choose a $w_n^t \sim P_n^t$ and send packets accordingly
6:         Observe feedback and calculate utility $u_n^t$ for the chosen $w_n^t$
           // Update each meta-distribution
7:         for $w \in W_n$ do
8:             Calculate $\hat{X}_{w,w'}^t, \forall w' \in W_n$ based on (6)
9:             $\hat{S}_{w,w'}^t = \hat{S}_{w,w'}^{t-1} + \hat{X}_{w,w'}^t, \forall w' \in W_n$
10:            Calculate $Q_w^{t+1}$ based on (7)
```

The LUC algorithm is described in Alg. 1. At beginning, all the meta-distributions are set with $q_{w,w'}^1 = \frac{1}{C_n}$, as we know little about the utilities for each *cwnd* (or *srate*) in advance. Then, initialize $\hat{S}_{w,w'}^0$ to be zero for all $w, w' \in W_n$, as shown in Line 2. Lines 4 to 6 show the process of calculating the distribution and observing the utility. The update for meta-distributions is described in Lines 7 to 10.

V. Analytical Results

A. Regret Bound

Recall that C_n is the number of actions in W_n. Then, the swap regret defined in (4) for a flow playing the LUC algorithm is bounded by the following theorem.

Theorem 1. *Let P_0 be a uniform distribution, and let $\delta \in (0, 1)$, $\beta = \sqrt{\frac{\ln(2C_n\delta^{-1})}{T}}$, $\eta = 0.25\sqrt{\frac{\ln C_n}{T}}$, and $\lambda = 0.5 C_n \sqrt{\frac{\ln C_n}{T}}$. When $T \geq C_n^2$, the cumulative swap regret for flow n playing the LUC algorithm over T rounds is upper bounded by $O(C_n\sqrt{T \log(C_n/\delta)})$ with probability at least $1 - \delta$.*

Proof Sketch. We need to prove the instantaneous regret bound for the swap regret, which is a stronger notion than the expected regret bound. However, the existing analysis techniques for the swap-regret-minimizing framework by [24] are only for the expected regret bound. Thus, we give a novel martingale-based analysis to derive a high-probability bound for any instantaneous actions and rewards.

In addition, we take the randomness of all flows into consideration. Although the proof is for each agent, the martingale sequence constructed in the proof is with respect to the past history of all flows. In the following, we omit subscript n in some notations for brevity.

The proof starts with a key step to decompose the swap regret bound as follows for any swap function $F \in \mathcal{F}$ such that $F(w) \in W_n$:

$$\max_{F\in\mathcal{F}} \sum_{t=1}^{T} \sum_{w\in W_n} \mathbf{1}[w_n^t = w] x_{F(w)}^t - \sum_{t=1}^{T} \sum_{w\in W_n} \mathbf{1}[w_n^t = w] x_w^t$$
$$= \max_{F\in\mathcal{F}} \left[\sum_{t=1}^{T} \sum_{w\in W_n} \mathbf{1}[w_n^t = w] x_{F(w)}^t - \sum_{t=1}^{T} \sum_{w\in W_n} p_w^t x_{F(w)}^t \right]$$
$$+ \left[\sum_{t=1}^{T} \sum_{w\in W_n} p_w^t x_{F(w)}^t - \sum_{t=1}^{T} \sum_{w\in W_n} \sum_{w'\in W_n} q_{w,w'}^t \hat{X}_{w,w'}^t \right]$$
$$+ \sum_{t=1}^{T} \left[\sum_{w\in W_n} \sum_{w'\in W_n} q_{w,w'}^t \hat{X}_{w,w'}^t - \mathbf{1}[w_n^t = w] x_w^t \right], \tag{8}$$

where we have decomposed the regret into three groups of terms.

For the first group of terms, we construct a martingale difference sequence, and apply Azuma's inequality to bound the term with a high probability.

For the second group of terms, we need to further decompose $q_{w,w'}^t$ into the combination of $\hat{q}_{w,w'}^t := \frac{\exp(\eta \hat{S}_{w,w'}^t)}{\sum_{w''\in W_n} \exp(\eta \hat{S}_{w,w''}^t)}$ and P_0. The key to bounding this term is to utilize the updating property for $\hat{q}_{w,w'}^t$ to bound the difference between $\sum_{t=1}^{T} \sum_{w\in W_n} \sum_{w'\in W_n} q_{w,w'}^t \hat{X}_{w,w'}^t$ and $\sum_{t=1}^{T} \sum_{w\in W_n} \max_{w'\in W_n} \hat{X}_{w,w'}^t$. Then, by constructing a super-martingale difference sequence, we can further bound the whole group of terms.

The third group of terms can be easily bounded by utilizing the property that $p_{w'}^t := \sum_{w\in W_n} p_w^t q_{w,w'}^t$. □

Note that Theorem 1 works for each flow playing the LUC algorithm. As each flow can be regarded as playing the adversarial multi-armed bandit problem [48], we can compare our results with the one in [48]. The key part of the analysis is to bound the regret with the swap functions instead of a fixed action in the external-regret analysis in [48], which guarantees the convergence to correlated equilibrium as discussed in Theorem 2. Furthermore, the analysis of our regret bound considers the randomness of all agent's actions. More importantly, we give a high-probability bound for the instantaneous swap regret, which is stronger than the expected regret studied in the literature.

If considering the time-averaged swap regret, LUC has an upper bound for the time-averaged swap regret of $O(\frac{C_n\sqrt{\log C_n}}{T})$. This means if playing LUC for a long time, both the external regret and internal regret are minimized, i.e., the time-averaged external regret and internal regret vanish asymptotically.

Regarding the optimality of the convergence rate, although our upper bound has a gap of $O(\sqrt{C_n})$ from the lower bound [25] of $\Omega(\sqrt{TC_n \log C_n})$, it is the best result so far for the exponential-weighting-based algorithm with the utility ranging in $[0, 1]$. However, the lower bound in [25] is for the full-information feedback, and it still remains an open problem whether the upper bound is tight or the lower bound is tight for the bandit feedback.

B. Convergence to Correlated Equilibria

Theorem 2 shows that the system can converge to an ϵ-correlated equilibrium if every flow in the network plays the LUC algorithm, by using the fact that the time-averaged internal-regret of each flow vanishes asymptotically over rounds.

Theorem 2. *If every flow $n \in \mathcal{N}$ plays the LUC algorithm for T rounds, then the empirical distribution of the joint actions played by all flows $\hat{\mathbf{P}}^T$ is an ϵ-correlated equilibrium with probability at least $1-\delta$ for $\delta \in (0,1)$.*

Proof. As explained in Sec. III, the swap regret can also boil down to the internal regret. By the swap-regret bound proved in Theorem 1 and the union bound over all flows, LUC has the internal regret for each flow n bounded for any action pairs $w, w' \in W_n$ with the probability at least $1-\delta$ for any $\delta \in (0,1)$ and $n \in \mathcal{N}$:

$$\sum_{t=1}^{T} r_{(w,w'),n}^t \le O(\max_{n\in\mathcal{N}} C_n \sqrt{T \log(C_n N/\delta)}).$$

Dividing both sides by T, and recalling that $\hat{\mathbf{P}}^T(\mathbb{W}) := \frac{1}{T} \sum_{t=1}^{T} \mathbf{1}\left[\mathbb{W}_t = \mathbb{W}\right], \mathbb{W} \in \mathcal{W}$, we have for any flow $n \in \mathcal{N}$ playing LUC and for any $w, w' \in W_n$

$$\sum_{\mathbb{W}: w_n = w} \hat{\mathbf{P}}(\mathbb{W}) \left(u_n\left(w'; \mathbb{W}_{-n}\right) - u_n\left(\mathbb{W}\right)\right) = O(\max_{n\in\mathcal{N}} C_n \sqrt{\frac{\log(C_n N/\delta)}{T}}),$$

which coincides with the definition of the ϵ-correlated equilibria in Sec. III by letting $\epsilon = O(\max_{n\in\mathcal{N}} C_n \sqrt{\frac{\log(C_n N/\delta)}{T}})$. □

The above theorem implies the existence of correlated equilibrium, as $\epsilon \to 0$ when $T \to \infty$. If every flow involved in the game plays the LUC algorithm together, the following corollary guarantees that the ϵ-correlated equilibrium can be found in a polynomial number of rounds in expectation.

Corollary 1. *Let $\delta \in (0,1)$. Then, with probability at least $1-\delta$, after $T = O(\max_{n\in\mathcal{N}} \frac{C_n^2 \log(C_n N/\delta)}{\epsilon^2})$ rounds, the empirical distribution $\hat{\mathbf{P}}^T$ of the joint actions played by all flows with the LUC algorithm is an ϵ-correlated equilibrium for the unknown general-sum game between multiple flows.*

Proof. By Theorem 2, with probability at least $1-\delta$, we can find an ϵ-correlated equilibrium for a game with N flows in T rounds. Then, solve the following equation for ϵ:

$$\epsilon = O(\max_{n\in\mathcal{N}} C_n \sqrt{\frac{\log(C_n N/\delta)}{T}}),$$

which gives $T = O(\max_{n\in\mathcal{N}} \frac{C_n^2 \log(C_n N/\delta)}{\epsilon^2})$. □

Thus, with the LUC algorithm played by all flows, correlated equilibria can be achieved with theoretical guarantees.

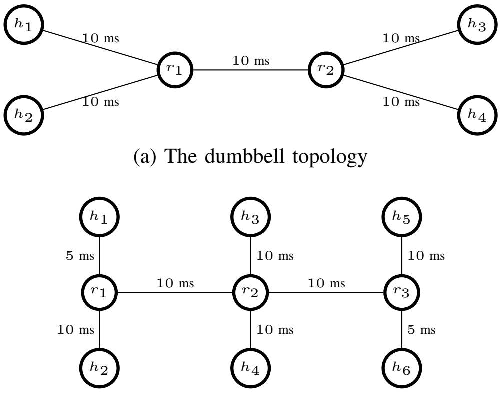
C. Time and Space Complexity

In each round, the most time-consuming parts are the calculation of the stationary distribution for the Markov chain, and the updating of C_n meta-distributions. Regarding the calculation of the stationary distribution for the Markov chain, each flow needs $O(C_n^2)$ time for C_n states [50]. Then, as for updating the meta-distributions, each meta-distribution needs $O(C_n)$ time to update C_n *cwnd* (or *srate*). Therefore, the time complexity of LUC is $O(C_n^2)$. Regarding the space complexity, each meta-distribution requires $O(C_n)$ space for C_n actions. Therefore, the space complexity for LUC is $O(C_n^2)$ for C_n meta-distributions.

As analyzed above, both the time and space complexities increase quadratically only with C_n, and are not affected by the scale of the network. Therefore, we can design a smaller action space for each flow in practice to make the algorithm more efficient. For example, converting the actions from the set of *cwnd* to the change of the *cwnd* (e.g., increase, decrease, or remain the same value).

VI. Experiments

We have implemented the proposed LUC algorithm with the same but normalized utility function defined in [20] through the Linux kernel 5.4.0 based on the congestion control plane [26], a new API for writing congestion control algorithms. In this section, we start with fairness-related experiments in Mininet [51], where LUC is compared with CUBIC [52] and BBR version 2 [53] (BBR2 for short in the following). Then, we conduct trace-driven experiments with Pantheon [27], an evaluation platform for congestion control algorithms, with an additional comparison to PCC-Vivace [20]. We have released the source code for the experiments in https://github.com/Zhiming-Huang/luc.



(a) The dumbbell topology

(b) The parking lot topology

Fig. 2: The experiment topology.

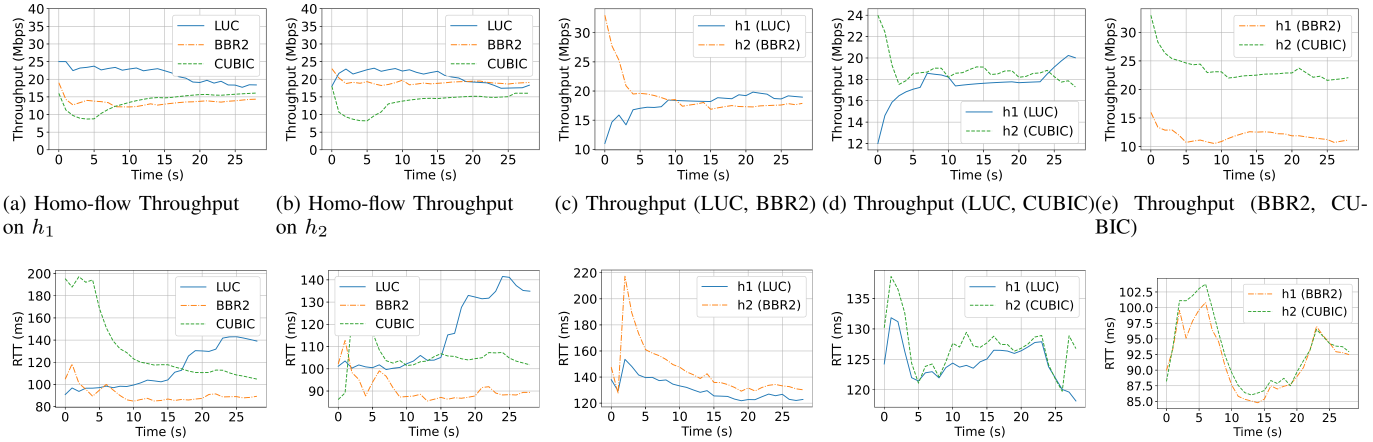
In the fairness-related experiments, two classic network topologies recommended by IETF TCP evaluation suite [54] are considered, i.e., the dumbbell and parking lot topologies, as shown in Fig. 2. The bandwidth of all the links in both topologies is 50 Mbps, and the delay of each link is shown in the figures. For both topologies, the queue size on all the links between two routers is 100 packets. In the dumbbell topology, there are two flows from h_1 to h_3 and from h_2 to h_4, sharing the same link r_1-r_2. In the parking lot topology, there are three flows from h_1 to h_6, from h_2 to h_3, and from h_4 to h_5. We can see that the flow from h_1 to h_6 competes with both the other two flows, while the other two flows are independent of each other. In the experiments, we use `iperf` to generate a 30s test for the performance of the three congestion control algorithms. As `iperf` outputs the averaged results (i.e., throughput and RTT) every 1s, the points at time 0s in Figs. 3 to 5 are the averaged results in the initial interval from 0s to 1s, and then we use exponentially weighted moving average technique to smooth the results in the following time.

In the trace-driven experiments, we use the US cellular network traces (i.e., T-mobile and Verizon) recorded by the saturator tool [55] while driving. These traces represent the time-varying capacity of the networks experienced by a mobile user, so we can test the adaptability of congestion control algorithms in such network environments.

A. Dumbbell Results

We first test the scenario where the two flows are homogeneous, i.e., both the flows adopt the same congestion control algorithm. The throughput and RTT results for the homogeneous flows are shown in Figs. 3a, 3b, 3f and 3g. As we can see, when both flows adopt LUC, they can achieve a fair performance, because LUC can guarantee a correlated equilibrium if played by all the flows. However, the other two congestion control algorithms, i.e., CUBIC and BBR2, cannot guarantee a fair share between the two flows. This is because of the intrinsic property of the two congestion control algorithms (i.e., deterministic strategy) and the subtle difference in the flow start time, although we have endeavored to minimize this difference in the experiments by using a script to control all nodes. The flow on h_1 slightly starts ahead of the flow on h_2, and thus the flow on h_1 with CUBIC or BBR2 will first occupy a bit more link bandwidth. Regarding RTT, we can see that LUC has a lower RTT than CUBIC. On the other hand, we can observe LUC performs better than BBR2 and CUBIC in the homogeneous setting in terms of throughput, while having a higher RTT. This is because LUC does not explicitly incorporate queuing models like BBR2 does, and the randomized action selection in LUC makes it less conservative when it comes to utilizing network buffers, leading to increased queue lengths.

We also conduct experiments for the heterogeneous flow, i.e., the two flows adopt different congestion control algorithms, as shown in Figs. 3c to 3e and Figs. 3h to 3j. When BBR2 and LUC compete with each other, BBR2 prevails at first but their gap gradually decreases due to the benefits of learning. On the other hand, LUC can achieve a similar throughput as CUBIC. Regarding RTT, we can observe in Fig. 3h that at the first few intervals, both LUC and BBR2 suffer a high RTT because they are competing with each other



(a) Homo-flow Throughput on h_1

(b) Homo-flow Throughput on h_2

(c) Throughput (LUC, BBR2)

(d) Throughput (LUC, CUBIC)

(e) Throughput (BBR2, CUBIC)

(f) Homo-flow RTT on h_1

(g) Homo-flow RTT on h_2

(h) RTT (LUC, BBR2)

(i) RTT (LUC, CUBIC)

(j) RTT (BBR2, CUBIC)

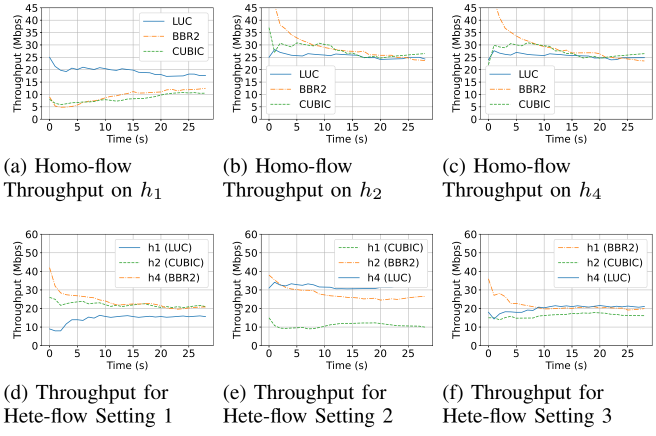
Fig. 3: The experiment results for the dumbbell topology.

to exhaust the network bandwidth. However, the RTTs for both algorithms are decreasing over time, meaning that the network is getting less congested. Therefore, LUC is friendly to and competitive with other TCP flows.

B. Parking Lot Results

The results of the parking lot topology are shown in Figs. 4 and 5. Similar to the dumbbell topology, we first test the homogeneous flows in the parking lot topology, as shown in Figs. 4a to 4c and Figs. 5a to 5c. The flow from h_1 to h_6 will suffer a loss if any one of the other two flows suffers, i.e., the flow from h_1 to h_6 has a higher probability of suffering a loss and thus results in a lower throughput than the other two flows. As we can see, when all the three flows play the same congestion control algorithm, LUC always guarantees that the performance gaps between flows are similar and not excessive. On the other hand, CUBIC and BBR2 have a large performance gap between flow h_1 to h_6 and the other two flows. This reflects LUC's ability to achieve a stable correlated equilibrium. Also, CUBIC and BBR2 incur a higher RTT than LUC, because CUBIC and BBR2 will increase the *srate* to probe for possible higher bandwidth, while LUC can maintain a low RTT.

Then, we perform three different heterogeneous flow settings in the parking lot topology for a different 4-hop flow (i.e., flow h_1 to h_6). In the first heterogeneous flow setting, flow h_1 to h_6 adopts LUC, flow h_2 to h_3 adopts CUBIC, and flow h_4 to h_5 adopts BBR2, and the results are shown in Figs. 4d and 5d. In the second setting, the 4-hop flow h_1 to h_6 adopts CUBIC, flow h_2 to h_3 adopts BBR2 and flow h_4 to h_5 adopts LUC, as shown in Figs. 4e and 5e. In the third setting, the 4-hop flow h_1 to h_6 would be BBR2, and the other two flows adopt CUBIC and LUC, respectively, as shown in Figs. 4f and 5f. We can see that when the 4-hop flow adopts LUC or CUBIC, it will concede the link bandwidth to the other two flows, but LUC still maintains a higher throughput than



(a) Homo-flow Throughput on h_1

(b) Homo-flow Throughput on h_2

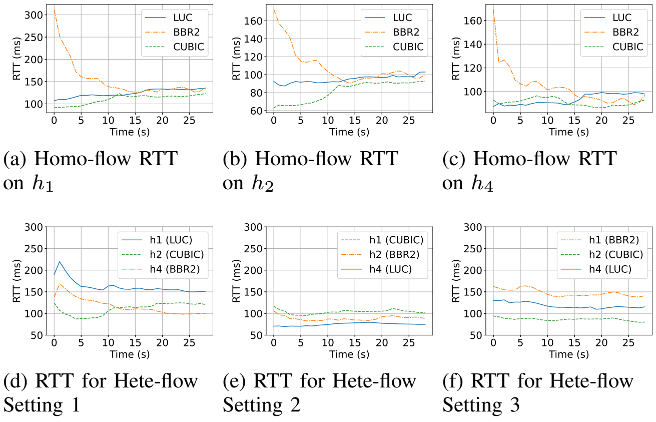
(c) Homo-flow Throughput on h_4

(d) Throughput for Hete-flow Setting 1

(e) Throughput for Hete-flow Setting 2

(f) Throughput for Hete-flow Setting 3

Fig. 4: The throughput results for the parking lot topology.



(a) Homo-flow RTT on h_1

(b) Homo-flow RTT on h_2

(c) Homo-flow RTT on h_4

(d) RTT for Hete-flow Setting 1

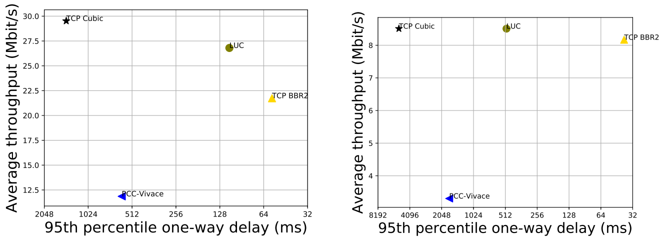
(e) RTT for Hete-flow Setting 2

(f) RTT for Hete-flow Setting 3

Fig. 5: The RTT results for the parking lot topology.

that of CUBIC, as shown in Figs. 4d and 4e. However, when the 4-hop flow adopts BBR2, it will still occupy a comparable link bandwidth with the other two flows, as shown in Fig. 4f. Overall, from the above experiments, we can see that LUC is friendly but competitive to other flows, and maintains fairness in allocating the link bandwidth.

C. Trace-driven Experiments



(a) T-mobile LTE Network

(b) Verizon LTE Network

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

For each trace, we did five independent runs of experiments, and the mean results for the trace-driven experiments produced by Pantheon are shown in Fig. 6. Pantheon evaluates an algorithm based on two metrics, i.e., mean throughput and 95th-percentile one-way delay. We can see that LUC can better balance the tradeoff between the average throughput and delay in these two metrics for the T-mobile network than the other three algorithms. In the Verizon LTE network, LUC still performs better than TCP CUBIC and Vivace. Although BBR2 is better than LUC in the Verizon LTE network, from the emulation results in Mininet, we know that the main advantage of LUC over BBR2 is the guarantee of fair allocation for multiple flows in the network. For both traces, PCC-Vivace performs not as well as other algorithms, as PCC-Vivace used by Pantheon runs in user space instead of kernel space, which may incur some performance loss. On the other hand, as Pantheon only supports evaluation for a single flow, our future work remains to conduct trace-driven emulations on multiple flows. Overall, LUC can guarantee good performance in a dynamic network environment.

VII. Conclusion

In this paper, we formulated the end-to-end congestion control as a repeated unknown general-sum game with bandit feedback, and have proposed the LUC algorithm with provable theoretical guarantees. Furthermore, we have implemented LUC through the Linux kernel and performed extensive experiments to verify the performance of LUC. For our future research, we would like to develop more realistic game models where we relax the assumption that all flows finish an interaction within one round, and study whether an equilibrium for such game models exists and can be obtained by efficient learning algorithms. We are also interested in using LUC as a building block to improve the current congestion control algorithms, such as BBR2.

Appendix

Let $\mathcal{F}_t := \sigma\left(\{\mathbb{W}_1, \ldots, \mathbb{W}_t\}\right)$ be the σ-algebra generated by previous actions of all agents by the end of round t. Then, denote by $\mathbf{E}_t[\cdot] := \mathbf{E}_t[\cdot \mid \mathcal{F}_{t-1}]$ the expectation conditioned on the $\mathcal{F}_{t-1}$. Let $x_w^t := u_n^t(w; \mathbb{W}_{-n}^t)$. Denote by $\hat{S}_w^T := \sum_{t=1}^{T} \sum_{w' \in W_n} q_{w,w'}^t \hat{X}_{w,w'}^t$ and $S_w^T := \sum_{t=1}^{T} \sum_{w \in W_n} \mathbf{1}[w_n^t = w] x_w^t$. As the proof is for a single agent n, we will omit subscript n in some notations for brevity.

Proof of Theorem 1. The instantaneous swap regret can be decomposed as follows

$$\begin{aligned}
&\max_{F \in \mathcal{F}} \sum_{t=1}^{T} \sum_{w \in W_n} \mathbf{1}[w_n^t = w] x_{F(w)}^t - \sum_{t=1}^{T} \sum_{w \in W_n} \mathbf{1}[w_n^t = w] x_w^t \\
&= \max_{F \in \mathcal{F}} \left[\sum_{t=1}^{T} \sum_{w \in W_n} \mathbf{1}[w_n^t = w] x_{F(w)}^t - \sum_{t=1}^{T} \sum_{w \in W_n} p_w^t x_{F(w)}^t \right] \\
&+ \left[\sum_{t=1}^{T} \sum_{w \in W_n} p_w^t x_{F(w)}^t - \underbrace{\sum_{t=1}^{T} \sum_{w \in W_n} \sum_{w' \in W_n} q_{w,w'}^t \hat{X}_{w,w'}^t}_{=:(\text{a})} \right] \\
&+ \sum_{t=1}^{T} \left[\underbrace{\sum_{w \in W_n} \sum_{w' \in W_n} q_{w,w'}^t \hat{X}_{w,w'}^t - \mathbf{1}[w_n^t = w] x_w^t}_{=:(\text{b})} \right],
\end{aligned} \tag{9}$$

First, notice that $\sum_{w \in W_n} \mathbf{1}[w_n^t = w] x_{F(w)}^t - \sum_{w \in W_n} p_w^t x_{F(w)}^t$ is a martingale difference sequence, because

$$\sum_{w \in W_n} \mathbf{E}_t \left[\mathbf{1}[w_n^t = w] x_{F(w)}^t \right] - \sum_{w \in W_n} \mathbf{E}_t \left[p_w^t x_{F(w)}^t \right] = \sum_{w \in W_n} \mathbf{E}_t \left[p_w^t x_{F(w)}^t - p_w^t x_{F(w)}^t \right] = 0$$

Furthermore, the martingale difference sequence is bounded:

$$\left| \sum_{w \in W_n} \mathbf{1}[w_n^t = w] x_{F(w)}^t - \sum_{w \in W_n} p_w^t x_{F(w)}^t \right| \leq 1,$$

where the inequality is due to that $\sum_{w \in W_n} \mathbf{1}[w_n^t = w] x_{F(w)}^t \in [0, 1]$ and $\sum_{w \in W_n} p_w^t x_{F(w)}^t \in [0, 1]$. Let $\delta' \in (0, 1)$. By applying Azuma's inequality, we have that with probability at least $1 - \delta'$,

$$\sum_{t=1}^{T} \sum_{w \in W_n} \mathbf{1}[w_n^t = w] x_{F(w)}^t - \sum_{t=1}^{T} \sum_{w \in W_n} p_w^t x_{F(w)}^t \leq 2\sqrt{T \ln \frac{1}{\delta'}}. \tag{10}$$

Then, we show how to bound (b) first as follows:

$$\begin{aligned}
(\text{b}) &= \sum_{w \in W_n} \sum_{w' \in W_n} q_{w,w'}^t \hat{X}_{w,w'}^t - \sum_{w \in W_n} \mathbf{1}[w_n^t = w] x_w^t \\
&= \sum_{w \in W_n} \sum_{w' \in W_n} \frac{p_w^t q_{w,w'}^t (\mathbf{1}[w_n^t = w'] x_{w'}^t + \beta)}{p_{w'}^t} - \sum_{w \in W_n} \mathbf{1}[w_n^t = w] x_w^t \\
&= \sum_{w' \in W_n} (\mathbf{1}[w_n^t = w'] x_{w'}^t + \beta) - \sum_{w \in W_n} \mathbf{1}[w_n^t = w] x_w^t = C_n \beta,
\end{aligned}$$

where the third equality is due to the definition of p_w^t in (5). Next, we show how to bound (a) as follows. Let $\hat{q}_{w,w'}^t := \frac{q_{w,w'}^t - \lambda P_0}{1-\lambda}$ be the distribution without mixing with P_0. Then, we obtain:

$$
\begin{aligned}
&-(\mathrm{a}) = -\sum_{w\in W_n}\sum_{t=1}^{T}\sum_{w'\in W_n} q_{w,w'}^t \hat{X}_{w,w'}^t = -\sum_{w\in W_n}\sum_{t=1}^{T}\sum_{w'}(1-\lambda)\hat{q}_{w,w'}^t\hat{X}_{w,w'}^t - \sum_{w\in W_n}\sum_{t=1}^{T}\sum_{w'}\lambda P_0 \hat{X}_{w,w'}^t \\
&\leq -\sum_{w\in W_n}\sum_{t=1}^{T}\sum_{w'}(1-\lambda)\hat{q}_{w,w'}^t\hat{X}_{w,w'}^t = \sum_{w\in W_n}\sum_{t=1}^{T}\frac{1-\lambda}{\eta}\underbrace{\left(\ln\sum_{w''\in W_n}\hat{q}_{w,w''}^t\exp\left(\eta(\hat{X}_{w,w''}^t - \sum_{w'\in W_n}\hat{q}_{w,w'}^t\hat{X}_{w,w'}^T)\right)\right)}_{=:(c)} \\
&-\sum_{w\in W_n}\sum_{t=1}^{T}\frac{1-\lambda}{\eta}\left(\ln\sum_{w''\in W_n}\hat{q}_{w,w''}^t\exp\left(\eta\hat{X}_{w,w''}^t\right)\right).
\end{aligned}
\tag{11}
$$

Then, notice that $q_{w,w'}^t \geq \lambda P_0 = \frac{\lambda}{C_n}$ and by the fact that $(1+\beta)\eta C_n \leq \lambda$, we further have that

$$
\eta\hat{X}_{w,w''}^t = \frac{\eta p_w^t q_{w,w''}^t(\mathbf{1}[w_n^t = w]x_w^t + \beta)}{p_{w''}^t q_{w,w''}^t} \leq \frac{p_w^t q_{w,w''}^t(\mathbf{1}[w_n^t = w]x_w^t + \beta)\eta C_n}{p_{w''}^t\lambda} \leq 1.
$$

Then, by using inequality $\ln x \leq x-1$ and $\exp(x) \leq 1+x+x^2$ for all $x \leq 1$, we have (c) bounded as follows:

$$
\begin{aligned}
(\mathrm{c}) &= \ln\sum_{w''\in W_n}\hat{q}_{w,w''}^t\exp\left(\eta\hat{X}_{w,w''}^t\right) - \eta\sum_{w'}\hat{q}_{w,w'}^t\hat{X}_{w,w'}^t \\
&\leq \sum_{w'\in W_n}\hat{q}_{w,w'}^t\exp\left(\eta\hat{X}_{w,w'}^t\right) - 1 - \eta\sum_{w'}\hat{q}_{w,w'}^t\hat{X}_{w,w'}^t \\
&\leq 1 + \sum_{w'\in W_n}\hat{q}_{w,w'}^t\eta\hat{X}_{w,w'}^t + \sum_{w'\in W_n}\hat{q}_{w,w'}^t(\eta\hat{X}_{w,w'}^t)^2 - 1 - \eta\sum_{w'}\hat{q}_{w,w'}^t\hat{X}_{w,w'}^t \\
&= \sum_{w'\in W_n}\frac{p_w^t\hat{q}_{w,w'}^t(\mathbf{1}[w_n^t = w]x_w^t + \beta)}{p_{w'}^t}\eta^2\hat{X}_{w,w'}^t \leq \sum_{w'\in W_n}\frac{1+\beta}{1-\lambda}\eta^2\hat{X}_{w,w'}^t,
\end{aligned}
$$

where the last inequality is due to that $\hat{q}_{w,w'}^t \leq \frac{q_{w,w'}^t}{1-\lambda}$. Substituting the above equation in (11) and recalling that $\hat{q}_{w,w'}^t = \frac{\exp\left(\eta\hat{S}_{w,w'}^{t-1}\right)}{\sum_{w''}\exp\left(\eta\hat{S}_{w,w''}^{t-1}\right)}$, we obtain for any $F \in \mathcal{F}$ that

$$
\begin{aligned}
&-(\mathrm{a}) \leq \sum_{w'\in W_n}\sum_{t=1}^{T}\sum_{w\in W_n}(1+\beta)\eta\hat{X}_{w,w'}^t - \sum_{w\in W_n}\sum_{t=1}^{T}\frac{1-\lambda}{\eta}\left(\ln\frac{\sum_{w'\in W_n}\exp\left(\eta\hat{S}_{w,w'}^t\right)}{\sum_{w''\in W_n}\exp\left(\eta\hat{S}_{w,w''}^{t-1}\right)}\right) \\
&\leq \sum_{w\in W_n}\sum_{t=1}^{T}\sum_{w'\in W_n}(1+\beta)\eta\hat{X}_{w,w'}^t - \sum_{w\in W_n}\frac{1-\lambda}{\eta}\left(\ln\left(\sum_{w'\in W_n}\exp\left(\eta\hat{S}_{w,w'}^T\right)\right) - \ln\left(\sum_{w'\in W_n}\exp\left(\eta\hat{S}_{w,w'}^0\right)\right)\right) \\
&\leq \sum_{w\in W_n}(1+\beta)\eta C_n\max_{w'\in W_n}\hat{S}_{w,w'}^t + \frac{1-\lambda}{\eta}\sum_{w\in W_n}\ln C_n - \frac{1-\lambda}{\eta}\sum_{w\in W_n}\ln\left(\exp\max_{w'\in W_n}\eta\hat{S}_{w,w'}^T\right) \\
&\leq \sum_{w\in W_n}((1+\beta)\eta C_n + \lambda)\max_{w'\in W_n}\hat{S}_{w,w'}^t + \frac{C_n\ln C_n}{\eta} - \sum_{w\in W_n}\max_{w'\in W_n}\hat{S}_{w,w'}^t \\
&\leq -(1-(1+\beta)\eta C_n - \lambda)\sum_{w\in W_n}\max_{w'\in W_n}\hat{S}_{w,w'}^t + \frac{C_n\ln C_n}{\eta} - \sum_{w\in W_n}\max_{w'\in W_n}\hat{S}_{w,w'}^t \\
&\leq -(1-(1+\beta)\eta C_n - \lambda)\sum_{w\in W_n}\hat{S}_{w,F(w)}^t + \frac{C_n\ln C_n}{\eta} - \sum_{w\in W_n}\max_{w'\in W_n}\hat{S}_{w,w'}^t,
\end{aligned}
\tag{12}
$$

where the last inequality is due to that $(1+\beta)\eta C_n \leq \lambda \leq \frac{1}{2}$. Notice that for any $w, w' \in W_n$, and by the fact that $\exp(x) \leq 1+x+x^2$ for $x \leq 1$ we have that

$$
\begin{aligned}
&\mathbf{E}_t\exp\left(\beta p_w^t x_{w'}^t - \beta\hat{X}_{w,w'}^t\right) = \mathbf{E}_t\exp\left(p_w^t\beta x_{w'}^t - \beta\frac{\mathbf{1}[w_n^t = w']p_w^t x_{w'}^t}{p_{w'}^t}\right)\cdot\exp\left(-\frac{\beta^2}{p_{w'}^t}\right) \\
&\leq \left(1 + \mathbf{E}_t\left[p_w^t\beta x_{w'}^t - \beta\frac{\mathbf{1}[w_n^t = w']p_w^t x_{w'}^t}{p_{w'}^t}\right] + \mathbf{E}_t\left[p_w^t\beta x_{w'}^t - \beta\frac{\mathbf{1}[w_n^t = w']p_w^t x_{w'}^t}{p_{w'}^t}\right]^2\right)\cdot\exp\left(-\frac{\beta^2}{p_{w'}^t}\right) \\
&\leq \left(1 + \frac{\beta^2}{p_{w'}^t}\right)\cdot\exp\left(-\frac{\beta^2}{p_{w'}^t}\right) \leq 1,
\end{aligned}
$$

where the last inequality is due to $1+x \leq \exp(x)$. Thus, we have that

$$
\mathbf{E}\left(\beta\sum_{t=1}^{T}p_w^t x_{w'}^t - \sum_{t=1}^{T}\beta\hat{X}_{w,w'}^t\right) \leq 1,
$$

and by Markov inequality, we obtain with probability at least $1-\delta'$ that

$$
\sum_{t=1}^{T}p_w^t x_{w'}^t - \sum_{t=1}^{T}\hat{X}_{w,w'}^t \leq \beta^{-1}\ln\frac{1}{\delta'},
$$

where $\delta' \in (0,1)$. Then, by union bound, we can continue to bound (12) with probability at least $1-\delta$ for any $\delta \in (0,1)$ as follows:

$$
\begin{aligned}
-(a) &\leq -(1-(1+\beta)\eta C_n - \lambda)\left(\sum_{w\in W_n}\sum_{t=1}^{T}p_w^t x_{F(w)}^t - \sum_{w\in W_n}\beta^{-1}\ln\frac{(C_n))^2}{\delta}\right) + \frac{C_n\ln C_n}{\eta} \\
&\leq ((1+\beta)\eta C_n + \lambda)T + \frac{C_n\ln C_n}{\eta} + 2C_n\beta^{-1}\ln\frac{C_n}{\delta} - \sum_{t=1}^{T}\sum_{w\in W_n}p_w^t x_{F(w)}^t,
\end{aligned}
$$

where the last inequality is due to the fact that $(1+\beta)\eta C_n \leq \lambda \leq 1/2$, and $x_{F(w)}^t \leq 1$.

By using the union bound to combine (10), and substituting the above results into (9), we have with probability at least $1-\delta$ that

$$
\begin{aligned}
&\max_{F\in\mathcal{F}}\sum_{t=1}^{T}\mathbf{E}_t\sum_{w\in W_n}\mathbf{1}[w_n^t = w]x_{F(w)}^t - \sum_{t=1}^{T}\mathbf{E}_t\sum_{w\in W_n}\mathbf{1}[w_n^t = w]x_w^t \\
&\leq ((1+\beta)\eta C_n + \lambda)T + \frac{C_n\ln C_n}{\eta} + 2C_n\beta^{-1}\ln\frac{C_n}{\delta} + C_n T\beta + 2\sqrt{T\ln\frac{2}{\delta}}.
\end{aligned}
$$

The theorem follows by letting $\beta = \sqrt{\frac{\ln(2C_n\delta^{-1})}{T}}$, $\eta = 0.25\sqrt{\frac{\ln C_n}{T}}$, and $\lambda = 0.5C_n\sqrt{\frac{\ln C_n}{T}}$. □

References

[1] C. Papadimitriou, "Algorithms, Games, and the Internet," in *Proc. Annual ACM Symposium on Theory of Computing (STOC)*, 2001, pp. 749–753.

[2] S. J. Shenker, "Making Greed Work in Networks: A Game-theoretic Analysis of Switch Service Disciplines," *IEEE/ACM Transactions on Networking (TON)*, vol. 3, no. 6, pp. 819–831, 1995.

[3] A. Akella, S. Seshan, R. Karp, S. Shenker, and C. Papadimitriou, "Selfish Behavior and Stability of the Internet: A Game-theoretic Analysis of TCP," *ACM SIGCOMM Computer Communication Review*, vol. 32, no. 4, pp. 117–130, 2002.

[4] R. Garg, A. Kamra, and V. Khurana, "A Game-theoretic Approach Towards Congestion Control in Communication Networks," *ACM SIGCOMM Computer Communication Review*, vol. 32, no. 3, pp. 47–61, 2002.

[5] X. Gao, K. Jain, and L. J. Schulman, "Fair and Efficient Router Congestion Control," in *Proc. Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, 2004, pp. 1050–1059.

[6] L. López, G. del Rey Almansa, S. Paquelet, and A. Fernández, "A Mathematical Model for the TCP Tragedy of the Commons," *Theoretical Computer Science*, vol. 343, no. 1–2, pp. 4–26, 2005.

[7] M. Chiang, S. Low, D. Wei, A. Tang, M. Chiang, S. Low, D. Wei, and A. Tang, "Heterogeneous Congestion Control: Efficiency, Fairness and Design," in *Proc. IEEE International Conference on Network Protocols (ICNP)*, 2006, pp. 127–136.

[8] A. Tang, J. Wang, S. H. Low, and M. Chiang, "Equilibrium of Heterogeneous Congestion Control: Existence and Uniqueness," *IEEE/ACM Transactions on Networking (TON)*, vol. 15, pp. 824–837, 2007.

[9] C. Chung and E. Pyrga, "Stochastic Stability in Internet Router Congestion Games," in *Proc. International Symposium on Algorithmic Game Theory (SAGT)*. Springer, 2009, pp. 183–195.

[10] P. S. Efraimidis, L. Tsavlidis, and G. B. Mertzios, "Window-games between TCP Flows," *Theoretical Computer Science*, vol. 411, no. 31–33, pp. 2798–2817, 2010.

[11] A. Mishra, J. Zhang, M. Sim, S. Ng, R. Joshi, and B. Leong, "Conjecture: Existence of Nash Equilibria in Modern Internet Congestion Control." in *Proc. Asia-Pacific Workshop on Networking (APNet)*, 2021, pp. 37–42.

[12] P. Thaker, M. Zaharia, and T. Hashimoto, "Don't Hate the Player, Hate the Game: Safety and Utility in Multi-Agent Congestion Control," in *Proc. ACM Workshop on Hot Topics in Networks (HotNets)*, 2021, pp. 140–146.

[13] R. Karp, E. Koutsoupias, C. Papadimitriou, and S. Shenker, "Optimization Problems in Congestion Control," in *Proc. Annual Symposium on Foundations of Computer Science (FOCS)*. IEEE, 2000, pp. 66–74.

[14] W. Li, F. Zhou, K. R. Chowdhury, and W. Meleis, "QTCP: Adaptive Congestion Control with Reinforcement Learning," *IEEE Transactions on Network Science and Engineering (TNSE)*, vol. 6, no. 3, pp. 445–458, 2018.

[15] N. Jay, N. Rotman, B. Godfrey, M. Schapira, and A. Tamar, "A Deep Reinforcement Learning Perspective on Internet Congestion Control," in *Proc. International Conference on Machine Learning (ICML)*. PMLR, 2019, pp. 3050–3059.

[16] S. Emara, B. Li, and Y. Chen, "Eagle: Refining Congestion Control by Learning from The Experts," in *Proc. IEEE Conference on Computer Communications (INFOCOM)*. IEEE, 2020, pp. 676–685.

[17] S. Abbasloo, C.-Y. Yen, and H. J. Chao, "Classic Meets Modern: A Pragmatic Learning-based Congestion Control for The Internet," in *Proc. ACM Special Interest Group on Data Communication on the applications, technologies, architectures, and protocols for computer communication (SIGCOMM)*, 2020, pp. 632–647.

[18] Z. Xia, Y. Chen, L. Wu, Y.-C. Chou, Z. Zheng, H. Li, and B. Li, "A Multi-objective Reinforcement Learning Perspective on Internet Congestion Control," in *Proc. IEEE/ACM International Symposium on Quality of Service (IWQOS)*. IEEE, 2021, pp. 1–10.

[19] S. Emara, F. Wang, B. Li, and T. Zeyl, "Pareto: Fair Congestion Control with Online Reinforcement Learning," *IEEE Transactions on Network Science and Engineering (TNSE)*, 2022.

[20] M. Dong, T. Meng, D. Zarchy, E. Arslan, Y. Gilad, B. Godfrey, and M. Schapira, "PCC Vivace: Online-Learning Congestion Control," in *Proc. USENIX Symposium on Networked Systems Design and Implementation (NSDI)*, 2018, pp. 343–356.

[21] E. Even-Dar, Y. Mansour, and U. Nadav, "On the Convergence of Regret Minimization Dynamics in Concave Games," in *Proc. Annual ACM symposium on Theory of Computing (STOC)*, 2009, pp. 523–532.

[22] H. H. Nax, M. N. Burton-Chellew, S. A. West, and H. P. Young, "Learning in A Black Box," *Journal of Economic Behavior & Organization*, vol. 127, pp. 1–15, 2016.

[23] A. R. Karlin and Y. Peres, *Game Theory, Alive*. American Mathematical Soc., 2017, vol. 101.

[24] A. Blum and Y. Mansour, "From External to Internal Regret," *Journal of Machine Learning Research (JMLR)*, vol. 8, no. 6, 2007.

[25] S. Ito, "A Tight Lower Bound and Efficient Reduction for Swap Regret," *Proc. Advances in Neural Information Processing Systems (NeurIPS)*, vol. 33, pp. 18 550–18 559, 2020.

[26] A. Narayan, F. Cangialosi, D. Raghavan, P. Goyal, S. Narayana, R. Mittal, M. Alizadeh, and H. Balakrishnan, "Restructuring Endpoint Congestion Control," in *Proc. the conference of the ACM Special Interest Group on Data Communication (SIGCOMM)*, 2018, pp. 30–43.

[27] F. Y. Yan, J. Ma, G. D. Hill, D. Raghavan, R. S. Wahby, P. Levis, and K. Winstein, "Pantheon: The Training Ground for Internet Congestion-control Research," in *Proc. USENIX Annual Technical Conference (ATC)*, 2018, pp. 731–743.

[28] J. Nagle, "On Packet Switches with Infinite Storage," *IEEE Transactions on Communications (TOC)*, vol. 35, no. 4, pp. 435–438, 1987.

[29] T. Alpcan and T. Basar, "A Game-theoretic Framework for Congestion Control in General Topology Networks," in *Proc. IEEE Conference on Decision and Control (CDC)*, vol. 2. IEEE, 2002, pp. 1218–1224.

[30] G. G. Anagnostopoulos, "Bayesian Games on A Maxmin Network Router," in *Proc. Mini-Conference on Applied Theoretical Computer Science (MATCOS)*, 2010, pp. 29–34.

[31] K. Winstein and H. Balakrishnan, "TCP Ex Machina: Computer-generated Congestion Control," *ACM SIGCOMM Computer Communication Review*, vol. 43, no. 4, pp. 123–134, 2013.

[32] G. W. Brown, "Some Notes on Computation of Games Solutions," RAND Corp Santa Monica CA, Tech. Rep., 1949.

[33] J. Robinson, "An Iterative Method of Solving a Game," *Annals of Mathematics*, pp. 296–301, 1951.

[34] N. Cesa-Bianchi and G. Lugosi, *Prediction, Learning, and Games*. Cambridge University Press, 2006.

[35] Q. Cui, Z. Xiong, M. Fazel, and S. S. Du, "Learning in Congestion Games with Bandit Feedback," in *Proc. Advances in Neural Information Processing Systems (NeurIPS)*, 2022, pp. 523–532.

[36] T. Boyarski, A. Leshem, and V. Krishnamurthy, "Distributed Learning in Congested Environments with Partial Information," 2021. [Online]. Available: https://arxiv.org/abs/2103.15901

[37] S. Tan, Z. Fang, Y. Wang, and J. Lü, "An Augmented Game Approach for Design and Analysis of Distributed Learning Dynamics in Multiagent Games," *IEEE Transactions on Cybernetics*, 2022.

[38] P. Coucheney, B. Gaujal, and P. Mertikopoulos, "Penalty-regulated Dynamics and Robust Learning Procedures in Games," *Mathematics of Operations Research*, vol. 40, no. 3, pp. 611–633, 2015.

[39] J. Cohen, A. Héliou, and P. Mertikopoulos, "Learning with Bandit Feedback in Potential Games," in *Proc. Advances in Neural Information Processing Systems (NeurIPS)*, 2017, pp. 6372–6381.

[40] J. Bielawski, T. Chotibut, F. Falniowski, G. Kosiorowski, M. Misiurewicz, and G. Piliouras, "Follow-the-Regularized-Leader Routes to Chaos in Routing Games," in *Proc. International Conference on Machine Learning (ICML)*, vol. 139. PMLR, 18–24 Jul 2021, pp. 925–935.

[41] D. H. Mguni, Y. Wu, Y. Du, Y. Yang, Z. Wang, M. Li, Y. Wen, J. Jennings, and J. Wang, "Learning in Nonzero-Sum Stochastic Games with Potentials," in *Proc. International Conference on Machine Learning (ICML)*, vol. 139. PMLR, 18–24 Jul 2021, pp. 7688–7699.

[42] M. Min and R. Hu, "Signatured Deep Fictitious Play for Mean Field Games with Common Noise," in *Proc. International Conference on Machine Learning (ICML)*, vol. 139. PMLR, 18–24 Jul 2021, pp. 7736–7747.

[43] W. Wang, J. Han, Z. Yang, and Z. Wang, "Global Convergence of Policy Gradient for Linear-Quadratic Mean-Field Control/Game in Continuous Time," in *Proc. International Conference on Machine Learning (ICML)*, vol. 139. PMLR, 18–24 Jul 2021, pp. 10 772–10 782.

[44] Q. Xie, Z. Yang, Z. Wang, and A. Minca, "Learning While Playing in Mean-Field Games: Convergence and Optimality," in *Proc. International Conference on Machine Learning (ICML)*, vol. 139. PMLR, 18–24 Jul 2021, pp. 11 436–11 447.

[45] G. Palaiopanos, I. Panageas, and G. Piliouras, "Multiplicative Weights Update with Constant Step-Size in Congestion Games: Convergence, Limit Cycles and Chaos," *Proc. Advances in Neural Information Processing Systems (NeurIPS)*, vol. 30, pp. 5872–5882, 2017.

[46] W. Krichene, B. Drighès, and A. M. Bayen, "Online Learning of Nash Equilibria in Congestion Games," *SIAM Journal on Control and Optimization*, vol. 53, no. 2, pp. 1056–1081, 2015.

[47] X. Chen and B. Peng, "Hedging in Games: Faster Convergence of External and Swap Regrets," *Proc. Advances in Neural Information Processing Systems (NeurIPS)*, 2020.

[48] P. Auer, N. Cesa-Bianchi, Y. Freund, and R. E. Schapire, "The Nonstochastic Multiarmed Bandit Problem," *SIAM Journal on Computing*, vol. 32, no. 1, pp. 48–77, 2002.

[49] S. Hart and A. Mas-Colell, "A Simple Adaptive Procedure Leading to Correlated Equilibrium," *Econometrica*, vol. 68, no. 5, pp. 1127–1150, 2000.

[50] B. N. Feinberg and S. S. Chiu, "A Method to Calculate Steady-state Distributions of Large Markov Chains by Aggregating States," *Operations Research*, vol. 35, no. 2, pp. 282–290, 1987.

[51] B. Lantz, B. Heller, and N. McKeown, "A Network in A Laptop: Rapid Prototyping for Software-defined Networks," in *Proc. the ACM SIGCOMM Workshop on Hot Topics in Networks*, 2010, pp. 1–6.

[52] S. Ha, I. Rhee, and L. Xu, "CUBIC: a New TCP-friendly High-speed TCP Variant," *ACM SIGOPS Operating Systems Review*, vol. 42, no. 5, pp. 64–74, 2008.

[53] N. Cardwell, Y. Cheng, S. H. Yeganeh, P. Jha, Y. Seung, I. Swett, V. Vasiliev, B. Wu, and M. M. V. Jacobson, "BBR v2: A Model-based Congestion Control," Montreal, Tech. Rep., 2019. [Online]. Available: https://datatracker.ietf.org/meeting/105/materials/slides-105-iccrg-bbr-v2-a-model-based-congestion-control-00

[54] D. Hayes, D. Ros, L. L. Andrew, and S. Floyd, "Common TCP Evaluation Suite," IETF, Internet-Draft draft-irtf-iccrg-tcpeval-01, Jul. 2014. [Online]. Available: https://datatracker.ietf.org/doc/draft-irtf-iccrg-tcpeval/01/

[55] K. Winstein, A. Sivaraman, and H. Balakrishnan, "Stochastic Forecasts Achieve High Throughput and Low Delay over Cellular Networks," in *Proc. USENIX Symposium on Networked Systems Design and Implementation (NSDI)*, 2013, pp. 459–471.