# Byzantine-Fault-Tolerant Consensus via Reinforcement Learning for Permissioned Blockchain-Empowered V2X Network

Seungmo Kim, *Senior Member*, *IEEE*, and Ahmed S. Ibrahim, *Member*, *IEEE*

***Abstract*—Blockchain has been forming the central piece of various types of vehicle-to-everything (V2X) network for trusted data exchange. Recently, permissioned blockchains garner particular attention thanks to their improved scalability and diverse needs from different organizations. One representative example of permissioned blockchain is Hyperledger Fabric ("Fabric"). Due to its unique execute-order procedure, there is a critical need for a client to select an optimal number of peers. The interesting problem that this paper targets to address is the tradeoff in the number of peers: a too large number will degrade scalability while a too small number will make the network vulnerable to faulty nodes. This optimization issue gets especially challenging in V2X networks due to mobility of nodes: a transaction must be executed, and the associated block must be committed before the vehicle leaves a network. To this end, this paper proposes a mechanism for selecting an optimal set of peers based on reinforcement learning (RL) to keep a Fabric-empowered V2X network impervious to dynamicity due to mobility. We model the RL as a contextual multi-armed bandit (MAB) problem. The results demonstrate the outperformance of the proposed scheme.**

***Index Terms*—Connected Vehicles; Blockchain; Hyperledger Fabric; BFT; RL; MAB**

## I. Introduction

### A. Background

Vehicle-to-everything (V2X) communications are acknowledged to have a massive potential to significantly decrease the number of vehicle crashes, thereby reducing the number of associated fatalities [1]. Based on this benefit, the literature perceives V2X communications as the central piece in constitution of intelligent transportation system (ITS) for connected and autonomous vehicles (CAVs).

Meanwhile, the blockchain technology has been gaining widespread interest based on its capability of providing secure, access-regulated interactions and transactions [2]. However, to be applied in V2X networks, the key challenge lies in keeping the performance of a consensus due to the networks' dynamicity attributed to mobility [3]. In general, a consensus algorithm is defined as a process to achieve agreement on a single data among distributed nodes. That is, a consensus algorithm is designed to achieve a certain degree of reliability even in a network involving unreliable nodes [4].

Permissioned blockchains are getting popular as a means to address this issue [5]. In many distributed blockchains, such as Ethereum and Bitcoin, which are not permissioned (also known as "public"), any node can participate in the consensus process, wherein transactions are ordered and bundled into blocks. Because of this characteristic, these systems rely on *probabilistic consensus* algorithms, which eventually guarantee consistency of a ledger. However, such a ledger still remains susceptible to divergence of ledgers (also known as a "fork"), where different participants in the network have a different view of the accepted order of transactions. Permissioned blockchains work differently. They aim at a *deterministic consensus* among all the nodes in a validation.

### B. Hyperledger Fabric

The Hyperledger Fabric ("Fabric" from now) has the widest popularity these days owing to its design as modular consensus protocols, which allows the system to be tailored to particular use cases and trust models [6]. It features existence of an entity called an orderer (also known as an "ordering node") that "commands" a consensus procedure for a transaction, which, along with other orderer nodes, forms an ordering service. Because Fabric's design adopts the deterministic consensus, any validated block is guaranteed to be final and correct.

Also, the Fabric features a unique *execute-order* architecture, which requires all peers to execute every transaction "before" the transaction is validated. Conversely, existing blockchain systems employ the opposite "order-execute" architecture: examples range from public blockchain such as Ethereum to permissioned ones adopted by various enterprises [7]. The limitation of the typical, order-execute architecture is apparent: every peer executes every transaction and transactions must be deterministic.

In a Fabric network, *scalability* is predominantly determined by the complexity of its endorsement policy [8] and ordering service where a consensus has to be reached [9]. Specifically, validation of endorsements on a transaction requires evaluation of an endorsement policy expression against the collected endorsements and checking for satisfiability [10], which is usually achieved via a *gossip protocol* in a consensus mechanism based on Byzantine fault tolerant (BFT). This is the key bottleneck in accomplishing scalability [11]: a larger number of peers participating in validation usually causes a longer latency and hence a lower throughput.

Moreover, there is a pitfall in the Fabric's execute-order structure [6]. Since an application is executed before validation of the associated transaction, the key drawback of this system occurs when the transaction turns out invalid at the end. It incurs a security problem and also waste of resources executing the application not complying the endorsement policy.

### C. Reinforcement Learning for Performance Optimization

In this paper, we propose to apply reinforcement learning (RL) to optimize the selection of an optimal set of peers

S. Kim is with the Department of Electrical and Computer Engineering, Georgia Southern University in Statesboro, GA, USA. Ahmed S. Ibrahim is with the Department of Electrical and Computer Engineering, Florida International University in Miami, FL, USA.

A preprint of this paper has been posted on arXiv.org, which can be found at https://arxiv.org/abs/2007.13957.

The work of S. Kim is supported in part by the Georgia Department of Transportation Grant No. RP 20-03, and that of Ahmed S. Ibrahim is supported in part by the National Science Foundation under Award No. CNS-1816112.

performing a consensus for a given block transaction in a Fabric network implemented in a V2X environment. However, there still remain challenges to address. Specifically, the learning is extremely complicated due to the dynamicity, which necessitates that the learning framework itself must be resilient and flexible according to the environment.

This paper proposes a learning mechanism formulated as a multi-armed bandit (MAB) problem, which enables a vehicle, without any assistance from an external infrastructure, to autonomously learn the environment and adapt its channel access behavior according to the outcome of the learning.

The MAB simplifies a RL by removing the learning dependency on *state* and thus providing evaluative feedback that depends entirely on the *actions*. The actions usually are decided upon in a greedy manner by updating the benefit estimates of performing each action independently from other actions. To consider the state in a bandit solution, *contextual bandits* may be used [13]. In many cases, a bandit solution may perform as well as a more complicated RL solution or even better. Many bandit algorithms feature stronger theoretical guarantees on their performance even under adversarial settings [14].

*Thompson sampling (TS)* (also known as *posterior sampling*) [27] provides a statistically efficient approach that tackles the exploration-exploitation dilemma by maintaining a posterior over models and choosing actions in proportion to the probability that they are optimal [15]. We will show in this paper that the endorsing peer selection problem can be solved via Thompson sampling.

### D. Contribution of This Paper

This paper proposes an endorser selection mechanism based on RL that is performed autonomously by a client. Specifically, this paper features the following contributions:

- It proposes an optimal channel selection protocol for the Fabric when applied to connected vehicles. This distinguishes our work from prior art, which mostly remains at simply adopting the Fabric in V2X networks without proposing a separate mechanism for performance optimization.
- It (i) adopts RL for accomplishing the aforementioned novel consensus protocol and (ii) models the optimization as a *contextual MAB* problem as means to achieve RL. Unlike the prior work such as [16] having provided only little technical details on the RL, this paper takes a more *balanced* view on both of the BFT and RL.
- It provides a *spatiotemporal* analysis framework for evaluating the performance of a blockchain system applied to a V2X network. The framework has advantages on several fronts: (i) the dynamics of vehicles are modeled by using stochastic processes; (ii) the time effects on the blockchain performance are evaluated; and (iii) the performance of RL is evaluated as Bayesian statistics.

## II. System Model

### A. Fabric-Based V2X Networking

*1) Key Terminologies for Fabric:* When "Fabric" is referred to in this paper, the fourth box from the top (named "Fabric network") in Fig. 2 will aid the reader's understanding.

This paper's system model implements three main components forming a Fabric network: namely, peer nodes ("*peers*" hereafter), channels and, organizations [7].

*Peers* are the key element of a Fabric network as they are the entities participating in a consensus procedure for validation of a block. One or a few of the peers are elected as the *orderer(s)* who act(s) as the commander of an ordering service. As such, a peer is the smallest unit of a member constituting a Fabric network.

A *channel* is defined as a mechanism via which peers interact with each other and with applications and exchange transactions privately. We emphasize the importance of understanding the concept of a channel in relation to a peer, since a channel is the object that the proposed mechanism targets to optimize, as shall be presented in Eq. (1). As such, a channel can be understood as a logical structure that is formed by a collection of peers: hence, peers provide the control point for access to, and management of, channels.

An *organization* is a party that has ownership and thus control of a Fabric network. A Fabric network is constituted of peers owned by the different organizations. For instance, a certain network established in a physical area can be governed by multiple parties: e.g., city, county, state, federal, and private enterprise. The Fabric allows a set of physical resources shared by multiple parties while each of the parties can maintain a private network built upon the resources, i.e., an organization.

Further, it is also worth to note of the mechanism that Fabric adopts to implement a consensus. The *Raft* is the protocol that Fabric uses for the consensus in its ordering service. Raft features a "leader and follower" model where a leader is dynamically elected among the ordering nodes in a channel, and that leader replicates messages to the follower nodes. Raft is considered a step forward to BFT from Kafka, its predecessor that was based on crash fault tolerant (CFT) [17].

*2) This Paper's Fabric-Based V2X Network Model:* This paper assumes a V2X network on which a permissioned blockchain is formed based on the *Fabric v2.0* [17]. Specifically, the roadside units (RSUs) act as *peers* that participate in endorsement and consensus (i.e., validation and commit) in a Fabric network, while the onboard units (OBUs)—i.e., vehicles—serve as *clients*. Applying the Fabric to this architecture, the RSUs have the authority to validate and order a block, which means that all the endorsing peers and orderers are selected from the RSUs. Meanwhile, OBUs request the execution and ordering services to RSUs.

By this architecture, we mean to make the blockchain system operate stably despite the vehicles' frequent entry into and departure from the blockchain network. Also, this architecture makes practical sense because a blockchain system will likely be managed by a certain party such as a state or federal organization or a private enterprise, through which vehicles pass and some of them may generate blocks that should be processed in the blockchain managed by such an organization.

As a significant remark, we remind that from v2.0, via an ordering service named "Raft," the Fabric started to provide a *BFT-based* consensus for validation and commit of a block. We emphasize that this also suits to address the dynamicity of a V2X network, which is highly dynamic in the network topology and the member composition, which implies a far higher possibility of malice or fault. As such, the employment of Fabric is justified in both aspects of efficiency and security.

*3) This Paper's Goal: Channel Selection Optimization:* As shall be detailed in Section III, the key problem statement of this paper is based on the tradeoff of channel selection. By definition, *channels* partition a Fabric network in such a way that only the stakeholders can view the transactions. In this way, organizations are able to utilize the same network while maintaining separation between multiple blockchains. The mechanism works by delegating transactions to different ledgers. Members of the particular channel can communicate and transact privately, while other members of the network cannot see the transactions on that channel. The Raft consensus service allows an orderer to *select* a channel through which it will serve the ordering service. As such, this paper focuses on *finding an optimal channel* that minimizes the latency and maximizes the throughput.

*4) Assumptions:* We assume that not all RSUs are connected to each other. A RSU usually has no wired connection, which causes that it only has a finite coverage [18]. Taking this practical aspect into consideration, we assume that only a certain number of RSUs falling into each other's communications range are connected. Interestingly, the Fabric does already consider this type of situation, which leads to employment of a *Gossip protocol* in disseminating information to reach a consensus during a block validation procedure.

It is also noteworthy that we consider a *discrete* time setting. Specifically, in each period $t = 1, \cdots, T$, where $T \in \mathbb{N}$ is a finite time horizon. It is a *synchronous* network [19], wherein all the clients (i.e., vehicles) and peers (i.e., RSUs) refer to the same discrete time $t$. As such, in the evaluation of this network's performance, we measure at any arbitrary node (i.e., a vehicle or a RSU) the number of slots that are consumed to process a transaction to append a block to the chain. We also remind to assume the *same length* of $t$ for all nodes.

## B. Geometry

We reiterate that not all nodes are connected directly to each other; however, every node is equipped with communications functionality and hence is able to exchange a transaction or a block to each other.

This paper adopts the stochastic geometry for characterization of a V2X network on a space [20]-[23]. They commonly rely on the fact that uniform distributions of nodes on $X$ and $Y$ axes of a Cartesian-coordinate two-dimensional space yield a Poisson point process (PPP) on the number of nodes in the space. The distributions of RSUs and OBUs are modeled as an independent homogeneous PPP $\Phi_r$ and $\Phi_o$ with the vehicle density $\lambda_r$ and $\lambda_o$.

A two-dimensional space $\mathbb{R}^2$ is defined with the length and width of $l$ and $w$ meters (m), respectively. To capture a more dynamic and realistic movement of nodes in a vehicular network, this system model considers *no separation of lanes*. Notice that such a generalized model enables the subsequent analyses more widely applicable [3]. Furthermore, to consider the most generic vehicle movement characteristic, this model assumes that every vehicle can move in any direction, which enables the system to capture every possible movement scenario including flight of unmanned aerial vehicles (UAVs), lane changing, intersection, and pedestrian walking.

## III. Proposed Mechanism

As was introduced in Section I-D, this paper proposes to enable a vehicle to (i) *autonomously learn* about a channel that provides an optimal number of peers and (ii) hence minimize the latency and maximize the throughput.

## A. Improvement to the Fabric Architecture

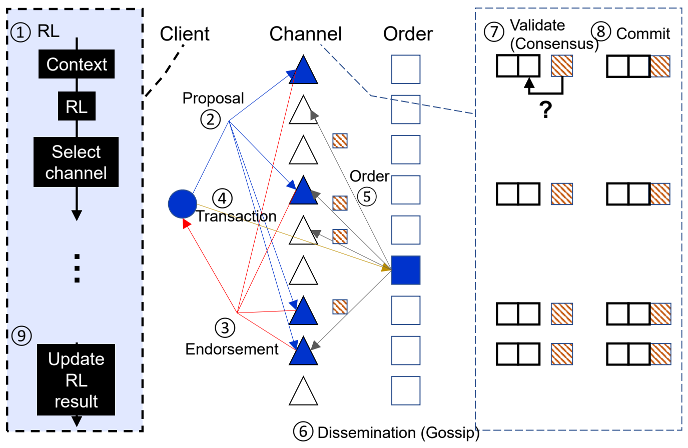
The key improvement is to introduce an optimal channel selection mechanism based on the RL. Fig. 1a demonstrates the *proposed RL-based execute-order mechanism*. We remind that in the proposed architecture, a vehicle becomes a client and RSUs serve as peers (i.e., endorsers and an orderer). A channel is formed among a certain subset of peers. There are multiple channels, from which a client can make a selection. Specifics of the procedure of the proposed RL-based consensus protocol is as follows: ① Each client has a training done before the beginning of joining a network for RL; ② When an application invokes, the client sends a proposal to the number of endorsers as a result of the RL; ③ The endorsers send the result of execution back to the client; ④ The client sends the endorsed transaction to an order; ⑤ the orderer puts the transaction into a block along with other transactions and multicasts the block to a set of endorsers that are directly connected to it; ⑥ The endorsers use a *gossip protocol* to disseminate the block to all among themselves; ⑦ The endorsers compare their ledgers if they are all final and validate the new block; ⑧ Once the endorsers reach a *consensus*, they append the block to the chain.

We emphasize that the proposed architecture is distinguished from the current Fabric's execute-order mechanism where a client works with only a certain subset of the peers (i.e., not all of the peers) depending on the endorsement policy in which the client operates [7]. Unless certain peers are designated by the policy, the client *randomly* selects the peers that will endorse its transaction. This is the part that this paper targets to improve: we propose a mechanism in which a client *learns* to optimize its selection of a channel.

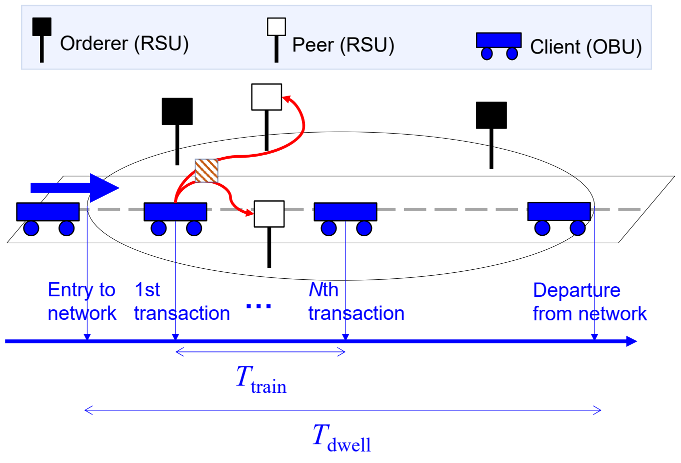
## B. RL for Channel Selection

*1) Spatiotemporal Perspective:* It is critical for a vehicle to collect the *prior* distribution for each channel. Fig. 1b illustrates a spatiotemporal view on a vehicle from its first entry into a Fabric network to departure. Before entry, the vehicle sends a Join Request (REQ) to the closest RSU, from which it receives a Join Confirm (CFM) upon entry to the network. The Join CFM message contains the information that is necessary to train a newly entering vehicle into the network: i.e., the minimum required number of endorsers and the latest number of clients queued in each endorser.

It is required that the vehicle $i$ needs to obtain the *prior distribution*, which will serve as the initial seed information



(a) Messaging process view



(b) Spatiotemporal view

Fig. 1: Proposed RL-based optimal channel selection mechanism

for a learning algorithm. However, a newly entering vehicle has *no prior information* about the channels available in the Fabric network. This explains the necessity of a dedicated time period for a newly entering vehicle to *train* itself order to make a decision that is close to an optimal. An arbitrary vehicle $i$ is designed to spend a certain length of time, $T_{\text{train}}$, observe the context $\mathbf{c}_i$ and update the reward $\mathbf{r}_i$. After $T_{\text{train}}$ elapses, the vehicle exploits the learned rewards among the arms.

*2) Problem Formulation:* Now, we present technical details of the MAB framework. Specifically, we characterize the proposed framework as a *contextual MAB problem* with the following details. That is, in time slot $t$, vehicle $i$ (i.e., a client for a blockchain) becomes an agent that observes the context and chooses an action based on the reward achieved from the action. Since the vehicle $i$ does not know an optimal action a priori, the vehicle needs to learn which action to select according to the given context and hence become able to optimize the transaction. In order to learn the policy, the vehicle has to try out different arms (i.e., channels defined in the current policy) for different contexts over time, which forms a contextual MAB problem. As such, in the proposed MAB problem, a newly entering vehicle (i.e., a client from the blockchain's point of views) is regarded a *bandit*, and each channel is modeled as an *arm* of the bandit.

Here is a justification for a "contextual" MAB. By definition, a bandit problem is defined as a single state version of a Markov decision process (MDP). However, in our proposed system, an action taken by a vehicle is affected by a context in which the vehicle lies in. Specifically, let us take the mobility, the key factor in determination of the scalability performance of the proposed mechanism. As described in Fig. 1b, our system model formulates the mobility by using the variable $T_{\text{dwell}}$, the length of time that a vehicle takes while passing through the Fabric network area. See Definition 1 for details on formulation of the context.

Now, we formulate our proposed mechanism into an optimization problem. Let $x_{i,t} \in \mathbb{R}^1$ denote the context of vehicle $i$ at time $t$. Also, we denote by $\mathsf{Y}_{i,t} \in \mathbb{R}^2$ a vector of possible actions. Notice that $\tilde{y}_{i,t} \in \mathsf{Y}_{i,t}$ is an action selected by the policy $\pi\left(y_{i,t}|x_{i,t}\right)$. Now, the goal is to train $\pi$ in order to maximize the reward $r\left(y_{i,t} \mid x_{i,t}\right)$ over a training period $T$ via a finite-horizon decision problem, aiming to find the optimal $\pi^*$, which yields an optimal action $\left(y_{i,t}\right)^*$. We formulate this process of predicting the optimal $\pi^*$, which is formally written as

$$\left(y_{i,t}\right)^* = \pi^*\left(y_{i,t} \mid x_{i,t}\right) = \underset{y_{i,t}\in\mathsf{Y}_{i,t}}{\operatorname{argmax}}\ r\left(y_{i,t} \mid x_{i,t}\right) \quad \text{subject to } c\left(y_{i,t} \mid x_{i,t}\right) \le C \tag{1}$$

where $c(\cdot)$ denotes the cost and $C$ gives the maximum acceptable cost for operating action $y_{i,t}$ in context $x_{i,t}$. Notice that in this problem setting, we define the cost as the *length of time taken for a consensus*, which is also called the *latency* as shall be shown in Fig. 6a.

**Remark 1** (0-1 knapsack problem). *We notice that the problem presented in Eq. (1) is a 0-1 knapsack problem (KP) [24], which aims to maximize the reward while keeping the cost under a certain level. The key challenge is that a KP already is a non-deterministic polynomial-time (NP)-complete problem [25], which will make prediction of $\pi$ cumbersome even with input variable $X$ in a low dimension. As a means to deal with this challenge, we take a numerical approach to produce the results, which will be presented in Section IV.*

The key challenge in a MAB problem lies in solving the *exploration vs. exploitation dilemma*, since all actions should be explored sufficiently often to learn their rewards, but also those actions which have already yielded high rewards should be exploited [26]. Further, an additional challenge unique to a "contextual" MAB problem is how to exploit historical *reward* observations under similar contexts. More technically, the problem of selecting an optimal channel comes from a tradeoff described in the following lemma:

**Lemma 1** (Tradeoff on the number of peers). *Regarding the constraint in Eq. (1), for a client, a tradeoff is formed in selecting a channel through which a transaction is executed* *and committed. In particular, the latency and throughput depend on "the number of peers." If there are too many peers, a higher latency will be caused for endorsement and consensus; on the other hand, too few peers will more easily cause a consensus failure when Byzantine faults occur.*

*3) Context:* Minimizing the need for modification to the current version of Fabric, we propose to design the *context* as those can be defined within an endorsement policy.

**Definition 1** (Context: Client's dwelling time in a Fabric network). *A client (i.e., a bandit in the MAB) makes an action based on its dwelling time in the Fabric blockchain network, which is denoted by $T_{dwell}$. The geographic information (e.g., the estimated radius of the network's boundary) is provided from the network via an endorsement policy in Join CFM upon joining of the network. Based on this information, each vehicle estimates its $T_{dwell}$ and uses it as the context to make the selection on a channel. It is formally written as $T_{dwell} = r/v$ where $r$ gives the radius of a Fabric network and $v$ denotes the speed of the tagged vehicle.*

*4) Reward:* We characterize this MAB as a "Beta-Bernoulli bandit" where the reward measured by vehicle $i$ in time $t$, $r_{i,t}$, is modeled to be either 1 (i.e., a success) or 0 (i.e., a failure).

**Definition 2** (Reward: Beta-Bernoulli bandit). *The reward for an action by client $i$ in time $t$ is defined as*

$$r_{i,t} = r_{i,t}^{\text{e\&c}} \cap r_{i,t}^{\text{l<d}} \tag{2}$$

*where*

$$r_{i,t}^{\text{e\&c}} = \mathbb{1}\,(\text{Block executed and committed to the chain})$$
$$r_{i,t}^{\text{l<d}} = \mathbb{1}\,(\text{Total latency shorter than } \mathsf{T}_{\text{dwell}})$$

*with $\mathbb{1}(\cdot)$ denoting an indicator function.*

As formulated above, the superscript "e&c" denotes an event where a transaction gets through both execution and commit, which means the transaction successfully completes Steps ① through ⑨ as described in Algorithm 1. Meanwhile, the superscript "l<d" indicates an event where a transaction can make it through Steps ① through ⑨ with a latency shorter than the vehicle's dwelling time $\mathsf{T}_{\text{dwell}}$ within the area of a Fabric network.

The *regret* of learning is defined as the difference between the reward achieved by vehicle $i$ in time slot $t$ and the optimal, which is formulated as

$$\rho_{i,t} = \left| r_{i,t} - r_{i,t}^{*} \right| \tag{3}$$

where $r_{i,t}^{*}$ denotes the reward that can be achieved by an optimal channel selection.

*5) Algorithm:* Now, we propose an *online learning algorithm* implementing the proposed contextual MAB problem. Notice that the algorithm is meant to run at each vehicle *on the fly* as the vehicle passes through an area operating a Fabric-based blockchain network. As described in Algorithm 1, the proposed framework features a RL mechanism to decide a channel, via which it sends a transaction proposal.

As shown in Line 4, a vehicle $i$ is recognized by a Fabric network upon its entrance to the network, which is certified

**Algorithm 1:** Proposed RL-based execution-order algorithm for a vehicle sending a block in Fabric-empowered V2X network

```
1  Input: $T_{\text{train}}$;   Output: $r_{i,t}$ and $(\alpha_k, \beta_k)$
2  Initialize: $\mathbf{r}_i, \mathbf{y}_i$
3  for $t = 1, \cdots, T_{dwell}$ do
4      Send Join REQ and receive Join CFM;
5      if $t \le T_{train}$ then
6          %— Training —%
7          $y_i \longleftarrow y_{i,t}$;
8          $\mathbf{r_i} \longleftarrow r_{i,t,k}$
9              where $r_{i,k} \sim \text{Beta}(\alpha_k, \beta_k) \longleftarrow (\alpha_k, \beta_k)_t$
10                                 $\forall k \in \{1, \cdots, \mathsf{N}_{\text{arm}}\}$;
11     else
12         %— Step ①: Channel selection —%
13         if $\epsilon$-greedy then
14             if rand $\le \epsilon$ then
15                 % Explore
16                 $\hat{k}_{i,t} = \mathcal{U}(\mathsf{N}_{\text{peer, min}}, \mathsf{N}_{\text{peer, max}})$;
17             else
18                 % Exploit
19                 $\hat{k}_{i,t} = \text{argmax}_k\, r_{i,k}\big|_{1,2,\cdots,t-1}$;
20             end
21         else
22             %— Thompson sampling —%
23             $\hat{\theta}_{i,t} \sim \text{Beta}(\alpha_k, \beta_k)$ for $k = 1, \cdots, \mathsf{N}_{\text{arm}}$;
24             $\hat{k}_{i,t} \longleftarrow \max_k \hat{\theta}_{i,t}$;
25         end
26         %— Steps ② and ③ —%
27         Send a transaction to the peers in channel $\hat{k}_{i,t}$;
28         Receive endorsement result from the peers in
             channel $\hat{k}_{i,t}$;
29         if Endorsement successful then
30             % Step ④
31             Request order;
32             (Steps ⑤-⑧ by orderers and endorsers)
33             if Validation successful then
34                 $r_{i,t}^{\text{e\&c}} \longleftarrow 1$;
35             else
36                 $r_{i,t}^{\text{e\&c}} \longleftarrow 0$;
37             end
38         else
39             $r_{i,t}^{\text{e\&c}} \longleftarrow 0$;
40         end
41         %— Latency examination —%
42         if Latency $\le T_{dwell}$ then
43             $r_{i,t}^{\text{l<d}} \longleftarrow 1$;
44         else
45             $r_{i,t}^{\text{l<d}} \longleftarrow 0$;
46         end
47         % Reward
48         $r_{i,t} \longleftarrow r_{i,t}^{\text{e\&c}} \cap r_{i,t}^{\text{l<d}}$;
49         % Step ⑨
50         $(\alpha_k, \beta_k) \longleftarrow (\alpha_k, \beta_k)_t + r_{i,t}$;
51     end
52 end
```

TABLE I: Parameters used for simulations

| Parameter | Setting |
|---|---|
| V2X networking | DSRC |
| Blockchain system | Hyperledger Fabric v2.0 |
| Consensus mechanism | BFT |
| Block dissemination for consensus | Gossip protocol |
| # channels | 10 |
| # peers per channel | Random ~ Uniform(5,10) |
| RL scheme | $\{\epsilon$-greedy, TS$\}$ |

by receiving a Join CFM message from the network admin server.

As described in Line 6, the vehicle should start a training period for $T_{\text{train}}$ slots upon new entry to a network. At time $t$, the vehicle $i$ observes context $c_{i,t}$ As a consequence, the vehicle collects the history of reward $r_i$ to update the *prior* distribution for the reward from channel $k$. Specifically, after $T_{\text{train}}$ elapses, for each arm $k$, the vehicle piles *successes* and *failures* to the prior, which is characterized as $r_{i,k} \sim \text{Beta}\left(\alpha_k, \beta_k\right)$.

From Line 11, now the vehicle $i$ starts to utilize the learned prior distribution to select a channel $\hat{k}_{i,t}$ when it needs to execute a transaction. As shown in Lines 13-23, a vehicle chooses between two representative strategies. In $\epsilon$-greedy, the vehicle still explores at the rate of $\epsilon$ and select channel $\hat{k}_{i,t}$ at random. At the other rate of $1-\epsilon$, it selects the $k$ having the greatest mean reward so far. Meanwhile, TS performs a sampling for each of the $\mathsf{N}_{\text{arm}}$ arms and selects channel $\hat{k}_{i,t}$ as $k$ showing the largest sample. We compare the prediction performance between the two learning schemes in Section IV.

Line 27 implements that upon selection of a channel $\hat{k}_{i,t}$, the vehicle $i$ is allowed to send a proposal to peers belonging to the channel. We remind that the proposal contains information about a new transaction that is generated by an application that the vehicle performs. In a Fabric network, the peers execute the application and simulates the transaction if it is valid as per the endorsement policy. If valid, each peer sends the vehicle an endorsement as shown in Line 28.

Upon collection of a sufficient number of endorsements, the vehicle now requests *validation* of the transaction to the orderer, which Line 31 describes. The next step is to examine whether the execution and commit have been successful, i.e., whether $r_{i,t}^{\text{e\&c}} = 1$, which is executed as in Lines 33 through 39. (We remind that this algorithm is designed to run on a vehicle; the tasks for orderers and endorsers—i.e., Tasks ⑤-⑧—are written in parentheses in Line 32.)

Now, as Lines 41 through 46 show, the vehicle examines if it has been able to receive a reply from the order confirming *commit* of the transaction while it still dwells in the network, the vehicle sets $r_{i,t}^{\text{l<d}} = 1$, or 0 otherwise. It is grave to notice that the condition in Line 42 is where the spatiotemporal analysis is taken into account. As reminded from Fig. 1b, $T_{\text{dwell}}$ measures how fast a vehicle passes through an orderer's communication range, which translates a spatial measure to a temporal one. We claim that this is how the algorithm holds the foundation for a spatiotemporal analysis, which therefore makes it capture the dynamicity of a V2X network.

Finally, the reward $r_{i,t}$ is computed and updated to the prior for each channel $k$, which are performed as Lines 47 through 50 in Algorithm 1. As evident from Lines 47-50 of the algorithm, the direct *output* upon completion of the algorithm consists of (i) a reward at vehicle $i$ as of time $t$, $r_{i,t}$, and (ii) a cumulated pair of successes and failures for channel $k$, $(\alpha_k, \beta_k)$. We recall that $\alpha_k$ and $\beta_k$ denote a success and a failure from a commit/execution, respectively, which form the shape parameters of a beta distribution. We remind that this $(\alpha_k, \beta_k)$ is ultimately used to identify the optimal channel.

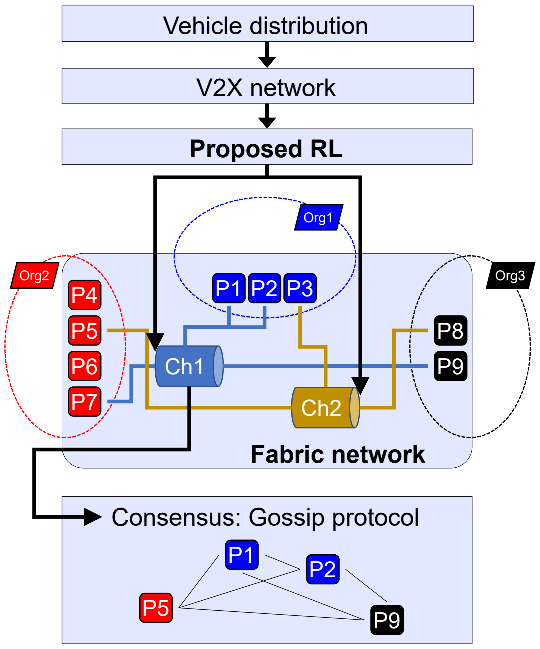


Fig. 2: The simulation software structure (An example with two channels, three organizations, and 9 peers)

## IV. Performance Evaluation

This section presents the results of numerical evaluation of the proposed framework, which are displayed in five metrics: namely, latency, block dissemination rate, convergence, scalability, and regret.

### A. Simulation Setting and Structure

For evaluating the performance, we constructed simulations for a Fabric-based V2X network on MATLAB. Table I summarizes the parameters and their settings. Our test Fabric network consists of three organizations, each of which is with 5-10 endorsing peers for a total of 100 peer nodes. We experimented on different numbers of channels: i.e., $\{10, 20, 30\}$ channels on different subsets of peers are tested. There is one orderer node operating in Raft [28].

Fig. 2 illustrates key components consisting of the simulation software and their structure. On the top, we generate vehicles on a *square*-shaped two-dimensional space $\mathbb{R}^2$ with the length and width of $l$ and $w$ m, respectively, as has been mentioned in Section II-B. The vehicles "dropped" on $\mathbb{R}^2$ operate based on the DSRC networking principles: i.e., 10 basic safety messages (BSMs) per second and listen-before-talk multiple access. The V2X network runs the Fabric overlaid on it and adopts the proposed RL-based algorithm to

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TIV.2022.3168575, IEEE Transactions on Intelligent Vehicles

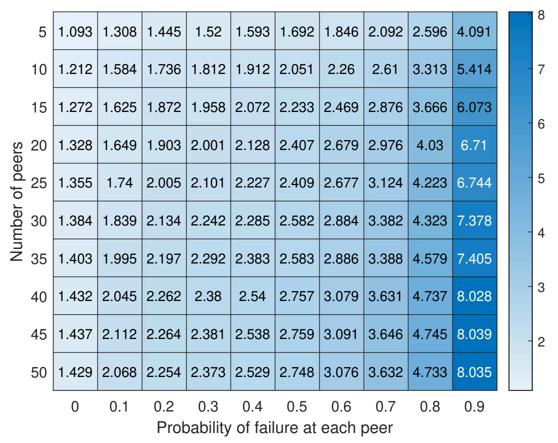


Fig. 3: Average latency (in seconds) vs. {number of peers ($\mathsf{N}_{\text{peer}}$), probability of fault ($p_f$)}

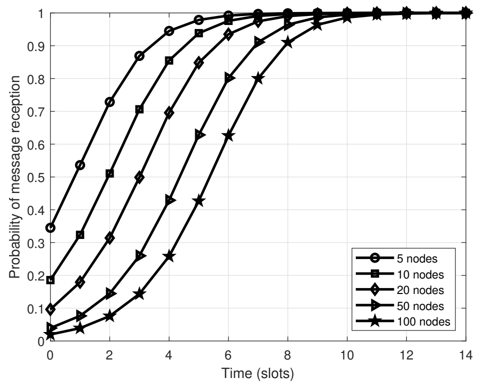


Fig. 4: Rate of dissemination of a block among a group of peers

optimize the performance of the Fabric network. As shown in the fourth block from the top, our simulation software implements several key components for a Fabric network: viz., organization, peer, channel, and orderer. Specifically, for each channel, we execute a Gossip protocol so the peers on the channel achieve a BFT-based consensus. The last block from the top shows an example of having Channel 1 selected, which is constituted by Peers {1, 2} belonging to Organization 1, Peer 5 belonging to Organization 2, and Peer 9 belonging to Organization 3.

## B. *Justification of Methodology*

We found that simulation would accomplish the best efficiency as the main method to evaluate the performance of the proposed mechanism, based on several advantages [4].

First, as shall be presented through Figs. 3 through 6, the parameters defining and operating the proposed mechanism are quite diverse in types and values, which makes it challenging to explore the parameters' dynamic orchestration in concert. A simulation provides a relatively easier control over such a large space composed of various parameters with wide ranges of values. It gives an obvious advantage over mathematical derivations and testbed implementations. An example is evaluation of a consensus process depending on the size of a channel. As presented in Table I, we intend to vary the number of channels in the range of $\mathsf{N}_{\text{ch}} = \{10, 20, 30\}$ with a different combination of peers for each channel every time a simulation is run. One can easily anticipate a high degree of complexity in changing the setting every time a new round of simulation is executed, while a large number of iterations is inevitable to present a statistically stable result in such a complex setting. As an effort to deal with the complexity, we adopt simulation as the main methodology, which, as shall be presented in this section, did efficiently evaluate the proposed system in a wide diversity of parameter settings.

Second, simulations enable computations without being caught up with restrictions or errors caused by computing environmental factors including hardware, compiler, language, etc. Accounting all the available options for all of those factors will complex the performance evaluation to a too high degree, which, thus, will make it hard to precisely identify the factors determining the key performance. In fact, existing literature has mentioned possible inaccuracy that could be caused by selection of a certain hardware [29]. As such, a *software suite* can provide a higher consistency than a real road test can do [30]. To wit, a simulation running on a computer provides an identical traffic scenario for multiple driver participants, which would likely be very costly and inaccurate in a road test due to the extreme variety in traffic situations including speed of other cars, road conditions, weather, etc.

With the above two advantages considered, we claim that when it comes to assessing the performance of a connected vehicle network, simulation is found to provide a more efficient approach compared to on-road experiment. We reiterate that a simulation can provide a lower-cost, easier-to-maneuver option to focus on a certain set of variables while keeping other variables controlled. Hence, the results that will be presented in this section are generated from simulations.

## C. *Results and Discussions*

Via Figs. 3 through 7, we demonstrate the results of the performance evaluation. In what follows, we discuss the implications of the results.

*1) Latency:* Fig. 3 shows the average latency versus (i) the number of peers in a channel, $\mathsf{N}_{\text{peer}}$, and (ii) the probability of fault, $p_f$, at each peer in a consensus procedure. Notice that the latency is defined as the total length of time that is taken for processing steps ① through ⑨ of the proposed mechanism. The degree of latency is expressed as the intensity on a "heatmap," as shown in the legend placed on the right-hand side of the heatmap.

To discuss what the result implies, it is obvious from the figure that a higher $p_f$ at each peer incurs a higher latency. The relationship is attributed to the fact that too many faulty peers can induce a delay in a consensus, which takes up a major part in the entire block validation process.

Conversely, it is interesting to observe that the dependency of the latency on $\mathsf{N}_{\text{peer}}$ is not monotonic. That is, a larger $\mathsf{N}_{\text{peer}}$ may incur a shorter latency, which suggests that an optimal $\mathsf{N}_{\text{peer}}$ yielding the smallest latency exists. The reason for this relationship is the tradeoff that was described in Lemma 1. To wit, too many peers take a longer time to achieve a consensus, while too few peers draw a higher chance of ending up with a consensus failure. In particular, if the latter is the case, it likely ends up with an even higher latency due to the need to start over the entire consensus process from the beginning.

*2) Block Dissemination Rate:* Figs. 4 shows the rate of dissemination of a block within a group of peers via a channel. Each curve indicates the length of time taken for propagation of a block from the master peer to all the other nodes. It is critical to recall that a consensus consumes the largest proportion in the latency for a block from being generated to being finally added to the chain. The dissemination latency ranges from 6 to 12 slots depending on the given numbers of peers–i.e., from 5 to 100.

To understand the result precisely, we analyze how a block is disseminated throughout a network via a Gossip protocol. Let $n$ denote the total number of nodes. Also, by $r_t$, we denote the proportion of nodes that have received the block sent from the source node after the execution of $t$ rounds. Meanwhile, $x_t$ gives the proportion of nodes that have not received the block yet: i.e., $x_t = 1 - r_t$. We assume that in the first time slot, i.e., $t_0$, the master peer has a block to propagate, which is given by $x_0 = 1/n$ and $r_0 = 1 - 1/n$. Now, for the propagation, we formulate the expectation of $x_t + 1$ as a function of $x_t$ as follows. Assuming uniform random selection of a node to receive the propagation in the next time slot $t$, the expected rate of reception of the block by the randomly selected node can be written as [32]

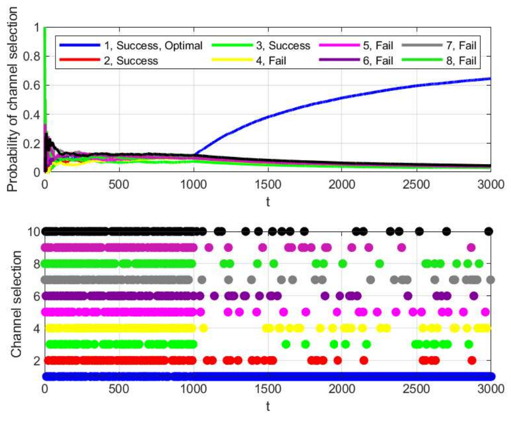
$$\mathbb{E}\left[r_{t+1}\right] = x_t \left(1 - \frac{1}{n}\right)^{n(1-x_t)}. \tag{4}$$

We remind that each curve in Fig. 4 describes $\mathbb{E}\left[r_{t+1}\right]$ versus $t$ for a value of $n$.

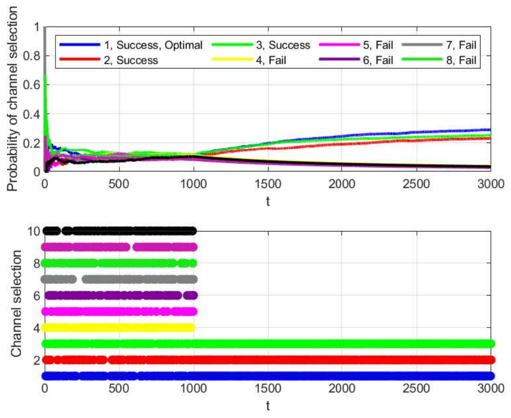
*3) Convergence:* To evaluate the time complexity, we compute the average length of time taken for selection of the optimal channel, depending on various values for $T_{\text{train}}$. Based on the fact that we model the MAB problem as a Bernoulli-bandit, we evaluate two representative algorithms finding an optimal arm in a MAB problem: Fig. 5 compares the convergence performance between $\epsilon$-greedy and TS.

Fig. 5 shows the results of an experiment where a vehicle learns on 10 channels. As an example, we set Channels 1, 2, and 3 as *successful* selections based on Definition 2, while Channel 1 is the optimum yielding the largest reward based on Eq. (2). Each of TS and $\epsilon$-greedy were run for $10^5$ iterations to demonstrate an average convergence performance.

The results reveal the following observation about the convergence of the learning algorithm. TS shows a better concentration on the eligible channels, but remains with sub-optimals (i.e., Channels 2 and 3) as well. Conversely, $\epsilon$-greedy still considers other irrelevant channels, yet it yields a higher probability of landing on the optimal channel. Specifically, while $\epsilon$-greedy can focus on a proved arm at the rate of 90% (since $\epsilon = 0.1$), it showed inefficiency by wasting time by still selecting irrelevant arms. On the other hand, TS is shown to better focus on the three successful arms as the learning progresses. In fact, TS has been evidenced to outperform other alternatives such as $\epsilon$-greedy and upper confidence bound (UCB) [31].



(a) $\epsilon$-greedy (With $\epsilon = 0.1$)



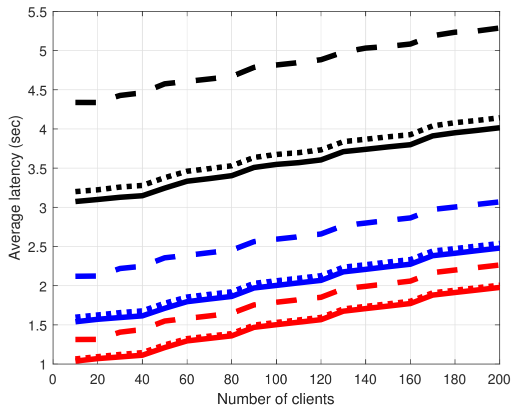
(b) TS

Fig. 5: Convergence of the proposed RL algorithm (With 10 channels; For each subfigure: Upper: Selected channel at each $t$, Lower: Probability of each channel selection over $t$)

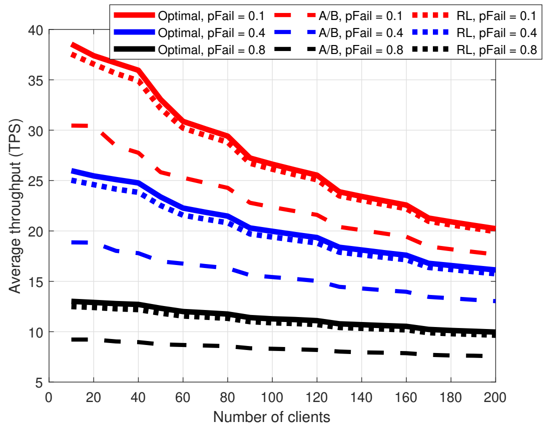
*4) Scalability:* Figs. 6a and 6b show the scalability via the latency and throughput versus the number of clients, as a result of the proposed RL mechanism applied in the proposed Fabric system framework for V2X. Notice of the definitions: *latency* is the time taken from application sending the transaction proposal to the transaction commit; and *throughput* is the rate at which transactions are committed to ledger, i.e., the number of transactions per second (TPS).

The key observation is that the proposed mechanism (shown as finely dotted lines in Fig. 6) achieves a performance that is far closer to the optimal than the current Fabric's channel selection mechanism. The rationale is the proposed RL scheme enables a vehicle to select a channel that provides a close-to-optimal number of peers, addressing the tradeoff that was described in Section III-B2.

*5) Regret:* Fig. 7 demonstrates the average regret according to the number of channels, the length of training period, and the method of RL—viz. $\epsilon$-greedy or TS. Notice that we refer to Eq. (3) for quantification of the regret. Comparing Figs. 7a and 7b suggests that TS results in a smaller regret as compared to $\epsilon$-greedy. The reason is that TS wastes a smaller number of trials for exploring arms with lower chances of winning than $\epsilon$-greedy does. Moreover, within each of Figs. 7a and 7b, it is

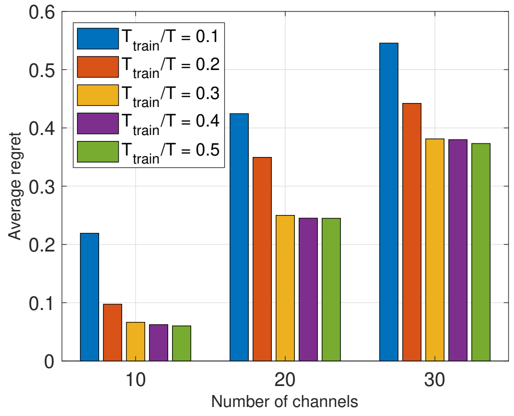


(a) Average latency vs. number of clients

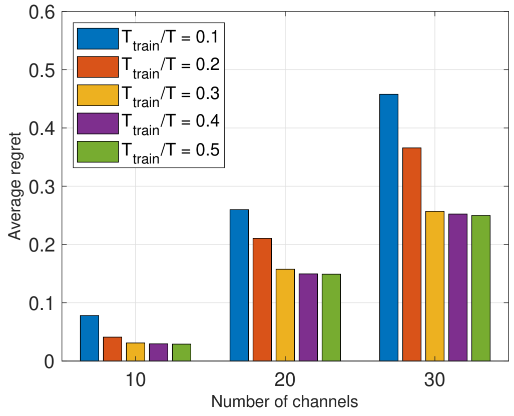


(b) Average throughput vs. number of clients

Fig. 6: Scalability



(a) $\epsilon$-greedy (With $\epsilon = 0.1$)



(b) TS

Fig. 7: Average regret vs. train time length

apparent that the regret is elevated with (i) a shorter training period and (ii) a larger number of channels to explore.

It is noteworthy that $T_{\text{train}}$ was left unitless for the following reason. We believe that the absolute length of $T_{\text{train}}$ is of less importance; rather, it has to be understood in reference to the length of an entire latency. For instance, $T_{\text{train}} = 1$ sec is a significant training burden when the entire blockchain process completes by $T = 2$ sec; yet, the same $T_{\text{train}} = 1$ sec is neglectable if the process takes $T = 100$ sec. This "*relativity*" is the main reason that we suggest displaying $T_{\text{train}}$ as a ratio to the total latency $T$.

Regardless, we would also like to emphasize that one can easily fathom the number of seconds for $T_{\text{train}}$ whenever the length $T$ is known. Notice that we define $T = T_{\text{dwell}}$ as evident from Line 3 of Algorithm 1. Also note that in reference to Fig. 1b, $T_{\text{dwell}}$ is a function of (i) the speed of a vehicle and (ii) the communication range of a RSU. That is to say, $T_{\text{dwell}}$ is an easy quantity to measure whenever one wants to know the value. It, in turn, makes $T_{\text{train}}$ equally easy to infer.

## V. Related Work

*1) V2X for ITS:* Security in vehicular networks is one of the most foremost aspects that have been pursued in the ITS literature [33]. A critical challenge in achieving security in a V2X network is the complexity and dynamicity attributed to mobility. A recent work in the literature proposed an optimal decision algorithm that is able to maximize the chance of making a correct decision on the message content, assuming the prior knowledge of the percentage of malicious vehicles in the network [34].

Meanwhile, the literature has also recognized blockchain as a main technological component to promote trust among vehicles. As an example, a privacy-improving blockchain architecture for smart vehicles has been proposed [35]. The mechanism features a blockchain mechanism enabling signatures to be exchanged without revealing the sender's identity, as a means to improve privacy.

However, none of the prior work has adequately addressed the key issue that this paper targets to discuss: the scalability for blockchain applied to a V2X network.

*2) Blockchain-Empowered V2X Network:* We found a body of prior work discussing blockchain applied on vehicular networks. An example is a RL-based industrial internet of things (IoT) [16]. Another RL-based performance optimization framework for blockchain-enabled internet of vehicles (IoV) was found, where transactional throughput is maximized while guaranteeing the decentralization, latency and security of the underlying blockchain system [36]. However, these existing methods limit its own applicability by assuming that a vehicle is able to select a certain consensus method. In practice, it is hard to switch the blockchain parameters in the middle of operating a consensus procedure.

Meanwhile, permissioned blockchains such as Fabric garnered considerable interest for application to ITS [7][37][38]. A key limitation from the literature is that no further details were discussed about optimization of selection of voting peers.

Another proposal focused on the endorsement procedure of Fabric [39]. It suggested an *anonymity* of endorsing peers in order to prevent a bias since revealing identity of each endorser among the peer nodes may not be suitable for transactions in which the endorsing peers have different preferences. However, we argue that not every application is biased; thus, it may incur unnecessary inefficiency if an application does not need anonymity. More importantly, the proposal limits its applicability to V2X since in many ITS applications (especially safety-critical ones), it is inappropriate to assume anonymous data exchange among vehicles.

In the current version of Fabric, a client can only *guess* in a selection of endorsing peers for a transaction [7]. Furthermore, the client application has no way of knowing which peers have updated ledgers and which do not, so the client may submit proposals to peers whose ledger data is not in sync with the rest of the network, keeping the transaction from being validated and thus committed.

As a remedy, the Fabric recently added the *service discovery* [40]. But it comes at the cost of a higher complexity due to the need for additional information that needs to be provided by each client. A scalability issue is still anticipated with a very large number of clients.

*3) Consensus in V2X Network:* The complexity of a BFT consensus in a vehicular network has been studied [42]. There was a recent proposal for vehicle-to-infrastructure (V2I) communications where the reputation is determined by the distance that a vehicle traveled [44]. Yet, the scope limited to the V2I channels, which has only little implication to a general V2X environment.

Another latest proposal proposed a scheme achieving distributed fault-tolerant consensus among connected vehicles [41]. However, our work features a RL-based algorithm finding an optimal number of peers participating a BFT consensus. Our algorithm is also sensitive to the latency of the algorithm in consideration to the dynamicity of a V2X network.

A BFT consensus algorithm was proposed for autonomous vehicles adopting the federated learning for privacy protection [43]. As compared to this prior art, our singular contribution is delving into the performance of a Fabric 2.0-empowered V2X network, which as such presents deeper technical specifics.

Focusing more on the Raft consensus that the Fabric adopts, albeit not many, there has been several latest proposals found in the literature. Examples include a BFT ordering service on top of a state machine replication/consensus library [45][46] and a grouped structure of Raft for stronger verification capability [47]. While they have relevant implication on how to make the Raft BFT, these prior proposals lack clarity on the applicability of the proposed methods to V2X environments.

*4) RL in V2X:* RL has been recently applied to wireless networks to provide a data-driven approach to solve traditionally challenging problems. Latest examples found in the literature include integration of networking, caching, and computing for connected vehicles [48]; safety optimization of finding trajectory for connected vehicles [49]; adaptive traffic signal control [50]; offloading for distributed computing [51].

We claim that this present paper applies the RL in optimization of the ordering service for a blockchain established among connected vehicles. That is where this paper distinguishes itself from the prior work.

## VI. Conclusions

This paper proposed a RL-based channel selection framework for the Fabric applied to V2X networks. We formulated the machine learning as a contextual MAB problem with the length of a vehicle's dwelling time in a Fabric network as the context. Specifically, we found that a tradeoff exists on the number of peers in a channel: a procedure of endorsement and consensus becomes (i) less scalable with too many peers and (ii) susceptible to faults with too few peers. Also, since the vehicle has no prior information of the peers' probability of fault upon joining a network, there is no way to anticipate the performance of each channel until it has learned about it. As an actual means to perform the learning, the proposed framework enabled a vehicle to adopt $\epsilon$-greedy and or TS. The results of our experiments showed that the proposed RL mechanism led to stable selection of channels fulfilling the success condition. More precisely, the proposed algorithm showed the latency and throughput close to the optimal.

This work is expected to have significant impact on future applications across the technologies gaining high research interest, namely Fabric and V2X. Despite advantages from its unique structure including modularization and execute-order procedure, the Fabric system still has many aspects to prove before stable operation in a V2X environment. One possible extension of this work is to incorporate the proposed RL mechanism to incorporate other dynamic factors such as network condition and evaluate the resulting performance impacts. It will also be interesting to apply this paper's structure to study the feasibility of other Hyperledger blockchains for V2X networks.

## References

[1] USDOT, "Vehicle-to-vehicle communication technology," *V2V Fact Sheet*, Jul. 2017. [Online]. Available: https://www.nhtsa.gov/sites/nhtsa.dot.gov/files/documents/v2vfactsheet101414v2a.pdf

[2] V. Sharma, "An energy-efficient transaction model for the blockchain-enabled Internet of vehicles (IoV)," *IEEE Commun. Lett.*, vol. 23, no. 2, Feb. 2019.

[3] S. Kim, "Impacts of mobility on performance of blockchain in VANET," *IEEE Access*, vol. 7, May 2019.

[4] A. Singh, T. Das, P. Maniatis, P. Druschel, and T. Roscoe, "BFT protocols under fire," in *Proc. USENIX NSDI 2008*.

[5] S. Namasudra, G. C. Deka, P. Johri, M. Hosseinpour, and A. H. Gandomi, "The revolution of blockchain: state-of-the-art and research challenges," *Springer Archives of Computational Methods in Engineering*, May 2020.

[6] M. Brandenburger, C. Cachin, R. Kapitza, and A. Sorniotti, "Blockchain and trusted computing: problems, pitfalls, and a solution for Hyperledger Fabric," *arXiv:1805.08541*, May 2018.

[7] E. Androulaki et al., "Hyperledger Fabric: A distributed operating system for permissioned blockchains," in *Proc. ACM EuroSys 2018*, Apr. 2018.

[8] P. Thakkar, S. Nathan, and B. Viswanathan, "Performance benchmarking and optimizing Hyperledger Fabric blockchain platform," in *Proc. IEEE MASCOTS 2018*.

[9] H. Sukhwani, N. Wang, K. S. Trivedi, and A. Rindos, "Performance modeling of hyperledger fabric (permissioned blockchain network)," in *Proc. IEEE NCA 2018*.

[10] J. Gottlieb, E. Marchiori, and C. Rossi, "Evolutionary algorithms for the satisfiability problem," *Evol. Comput.*, vol. 10, no. 1, Mar. 2002.

[11] C. Berger, H. P. Reiser, "Scaling byzantine consensus: A broad analysis," in *Proc. ACM Workshop on Scalable and Resilient Infrastructures for Distributed Ledgers (SERIAL) 2018*.

[12] S. Nakamoto, "Bitcoin: A peer-to-peer electronic cash system," Oct. 2008. [Online]. Available: https://bitcoin.org/bitcoin.pdf

[13] W. Chu, L. Li, L. Reyzin, and R. Schapire, "Contextual bandits with linear payoff functions," in *Proc. Int. Conf. Artificial Intell. Statist. 2011*.

[14] A. Haj-Ali, N. K. Ahmed, T. Willke, J. E. Gonzalez, K. Asanovic, and I. Stoica, "A view on deep reinforcement learning in system optimization," *arxiv:1908.01275v3*, Sep. 2019.

[15] C. Riquelme, G. Tucker, and J. Snoek, "Deep Bayesian bandits showdown," *arXiv:1802.09127v1*, Feb. 2018.

[16] M. Liu, F. R. Yu, Y. Teng, V. C. M. Leung, and M. Song, "Performance optimization for blockchain-enabled industrial internet of things (IIoT) systems: A deep learning reinforcement learning approach," in *IEEE Trans. Ind. Informat.*, vol. 15, no. 6, Jun. 2019.

[17] Website of the Hyperledger Fabric v2.0. [Online]. Available: https://www.hyperledger.org/blog/2020/01/30/welcome-hyperledger-fabric-2-0-enterprise-dlt-for-production

[18] Siemens Connected Vehicle RSU, [Online]. Available: https://www.mobotrex.com/product/siemens-connected-vehicle-roadside-unit/

[19] IEEE, *IEEE 802.11p-2010 - IEEE Standard for Information technology– Local and metropolitan area networks– Specific requirements–Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications Amendment 6: Wireless Access in Vehicular Environments*, Jun. 2010.

[20] S. Kim and C. Dietrich, "A novel method for evaluation of coexistence between DSRC and Wi-Fi at 5.9 GHz," in *Proc. IEEE Globecom 2018*.

[21] S. Kim and T. Dessalgn, "Mitigation of civilian-to-military interference in DSRC for urban operations," in *Proc. IEEE MILCOM 2019*.

[22] S. Kim and M. Bennis, "Spatiotemporal analysis on broadcast performance of DSRC with external interference in 5.9 GHz band," *arXiv:1912.02537*, Dec. 2019. [Online]. Avilable: https://arxiv.org/pdf/1912.02537.pdf

[23] S. Kim and B. J. Kim, "Novel backoff mechanism for mitigation of congestion in DSRC broadcast," *arXiv:2005.08921*, May 2020.

[24] K. W. Ross and D. H, K. Tsang, "The stochastic knapsack problem," *IEEE Trans. Commun.*, vol. 37, no. 7, Jul. 1989.

[25] V. Poirriez, N. Yanev, and R. Andonov, "A hybrid algorithm for the unbounded knapsack problem," *Discrete Optimization*, vol. 6, no. 1, Feb. 2009.

[26] D. J. Russo, B. Van Roy, A. Kazerouni, I. Osband, and Z, Wen, "A tutorial on Thompson sampling," *Foundations and Trends in Machine Learning*, vol. 11, no. 1, Jul. 2018.

[27] A. Slivkins, "Introduction to multi-armed bandits," *arXiv:1904.07272v5*, Sep. 2019.

[28] Hyperledger Fabric, "The ordering service," Docs, Release-2.2, [Online]. Available: https://hyperledger-fabric.readthedocs.io/en/release-2.2/orderer/ordering_service.html

[29] M. Brandenburger, C. Cachin, R. Kapitza, and A. Sorniotti, "Blockchain and trusted computing: Problems, pitfalls, and a solution for Hyperledger Fabric," *arXiv:1805.08541v1*, May 2018. [Online]. Available: https://arxiv.org/pdf/1805.08541.pdf

[30] R. A. Wynne, V. Beanlanda, and P. M. Salmon, "Systematic review of driving simulator validation studies," *Elsevier Safety Science*, vol. 117, Apr. 2019.

[31] O. Chapelle and L. Li, "An empirical evaluation of Thompson sampling," *Advances in Neural Inform. Process. Syst.*, 2011.

[32] M. Jelasity, *Gossip*, Springer Berlin Heidelberg, pp. 139–162, [Online]. Available: http://dx.doi.org/10.1007/978-3-642-17348-6_7

[33] Z. Lu, G. Qu, and Z. Liu, "A survey on recent advances in vehicular network security, trust, and privacy," *IEEE Trans. Intell. Transp. Syst.*, vol, 20, iss. 2, Feb. 2019.

[34] J. Chen, G. Mao, C. Li, and D. Zhang "A topological approach to secure message dissemination in vehicular networks," *IEEE Trans. Intell. Transp. Syst.*, vol. 21, iss. 1, Jan. 2020.

[35] L. Li, J. Liu, L. Cheng, S. Qui, W. Wang, X. Zhang, and Z. Zhang, "CreditCoin: A privacy-preserving blockchain-based incentive announcement network for communications of smart vehicles," *IEEE Trans. Intell. Transp. Syst.*, vol. 19, iss. 7, Jul. 2018.

[36] C. Qiu, F. R. Yu, F. Xu, H. Yao, and C. Zhao, "Blockchain-based distributed software-defined vehicular networks via deep Q-learning," in *Proc. ACM DIVANet 2018*.

[37] J. Gao, K. Agyekum, E. Sifah, K. Acheampong, Q. Xia, X. Du, M. Guizani, and H. Xia, "A blockchain-SDN-enabled internet of vehicles environment for fog computing and 5G networks," *IEEE Internet Things J.*, vol. 7, no. 5, Nov. 2019.

[38] W. Li, H. Guo, M. Nejad, and C. Shen, "Privacy-preserving traffic management: A blockchain and zero-knowledge proof inspired approach," *IEEE Access*, vol. 8, Oct. 2020.

[39] S. Mazumdar and S. Ruj, "Design of anonymous endorsement system in Hyperledger Fabric," in *IEEE Trans. Emerg. Topics Comput.*, Jun. 2019.

[40] Y. Manevich, A. Barger, and Y. Tock, "Service discovery for Hyperledger Fabric," *IBM J. Res. Dev*, vol. 63, Feb. 2019.

[41] T. Li, L. Tseng, T. Higuchi, S. Ucar, and O. Altintas, "Poster: Fault-tolerant consensus for connected vehicles: A case study," in *Proc. IEEE VNC 2021*.

[42] D. Rjazanovs, E. Petersons, A. Ipatovs, L. Juskaite, and R. Yeryomin, "Byzantine failures and vehicular networks," in *Proc. IEEE MTTW 2021*.

[43] J. Chen, M. Chen, G. Zeng, and J. Weng, "BDFL: A Byzantine-fault-tolerance decentralized federated learning method for autonomous vehicle," *IEEE Trans. Veh. Technol.*, vol. 70, no. 9, Sep. 2021.

[44] D. Suo, J. Zhao, and S. Sarma, "Proof of travel for trust-based data validation in V2I communication part I: Methodology," *arXiv preprint arXiv:2104.05070*, Apr. 2021.

[45] J. Sousa, A. Bessani, and M. Vukolic, "A byzantine fault-tolerant ordering service for the hyperledger fabric blockchain platform," in *Proc. IEEE/IFIP DSN*.

[46] A. Barger, Y. Manevich, H. Meir, and Y. Tock, "A Byzantine fault-tolerant consensus library for Hyperledger Fabric," *arXiv:2107.06922v1*, Ju. 2021.

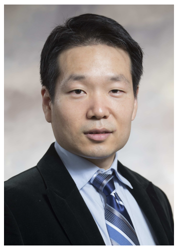
[47] S. Zhou and B. Ying, "VG-Raft: An improved Byzantine fault tolerant algorithm based on Raft algorithm," in *Proc. IEEE ICCT 2021*.

[48] Y. He, N. Zhao, and H. Yin, "Integrated networking, caching, and computing for connected vehicles: A deep reinforcement learning approach," *IEEE Trans. Veh. Technol.*, vol. 67, no. 1, Jan. 2018.

[49] T. Ghoul and T. Sayed, "Real-time safety optimization of connected vehicle trajectories using reinforcement learning," *MDPI Sensors*, vol. 21, 2021.

[50] W. Li, Z. Mo, Y. Fu, K. Ruan, X. Di, "CVLight: Decentralized learning for adaptive traffic signal control with connected vehicles," *arXiv:2104.10340v2*, Dec. 2021.

[51] C. Chen, Y. Zhang, Z. Wanga, S. Wan, Q. Pei, "Distributed computation offloading method based on deep reinforcement learning in ICV," *Applied Soft Comput. J.*, vol. 103, 2021.



**Seungmo Kim** (Senior Member, IEEE) received his B.S. and M.S. degrees in electrical communications engineering from Korea Advanced Institute of Science and Technology (KAIST) in Daejeon, South Korea, in 2006 and 2008, respectively, and the Ph.D. degree in electrical engineering from Virginia Tech, Blacksburg, VA, USA in 2017. Since 2017, he has been an assistant professor of Electrical and Computer Engineering at Georgia Southern University, Statesboro, GA, USA. His current research focus is on connected vehicles, blockchain, and reinforcement learning. He is a recipient of the best paper award at IEEE WCNC 2016, IARIA ICNS 2021, and IEEE ICEIC 2022.

**Ahmed S. Ibrahim** (Member, IEEE) received the B.S. degree (Hons.) and the M.S. degree in electronics and electrical communications engineering from Cairo University, Cairo, Egypt, in 2002 and 2004, respectively, and the Ph.D. degree in electrical engineering from the University of Maryland, College Park, MD, USA, in 2009. He was an Assistant Professor with Cairo University, a Research Scientist with Intel, and a Senior Engineer with Inter Digital Communications. He is currently an Assistant Professor of Electrical and Computer Engineering, Florida International University, Miami, FL, USA. His research interests include next-generation mobile communications, drone-assisted millimeter-wave communications, and vehicular networks.