Federated Learning With Blockchain for Autonomous Vehicles: Analysis and Design Challenges

Shiva Raj Pokhrel, *Member, IEEE*, and Jinho Choi, *Senior Member, IEEE*

***Abstract*— We propose an autonomous blockchain-based federated learning (BFL) design for privacy-aware and efficient vehicular communication networking, where local on-vehicle machine learning (oVML) model updates are exchanged and verified in a distributed fashion. BFL enables oVML without any centralized training data or coordination by utilizing the consensus mechanism of the blockchain. Relying on a renewal reward approach, we develop a mathematical framework that features the controllable network and BFL parameters (e.g., the retransmission limit, block size, block arrival rate, and the frame sizes) so as to capture their impact on the system-level performance. More importantly, our rigorous analysis of oVML system dynamics quantifies the end-to-end delay with BFL, which provides important insights into deriving optimal block arrival rate by considering communication and consensus delays. We present a variety of numerical and simulation results highlighting various non-trivial findings and insights for adaptive BFL design. In particular, based on analytical results, we minimize the system delay by exploiting the channel dynamics and demonstrate that the proposed idea of tuning the block arrival rate is provably online and capable of driving the system dynamics to the desired operating point. It also identifies the improved dependency on other blockchain parameters for a given set of channel conditions, retransmission limits, and frame sizes.[1] However, a number of challenges (gaps in knowledge) need to be resolved in order to realise these changes. In particular, we identify key bottleneck challenges requiring further investigations, and provide potential future reserach directions.**

***Index Terms*— On-vehicle machine learning, federated learning, blockchain, delay analysis, consensus delay, low delay.**

I. Introduction

NEXT-GENERATION wireless networks are envisaged to guarantee low-delay and ultra-high reliable connectivity anywhere, anytime and on-the-move [2], [3]. This will satisfy the real-time communication constraints for the impending autonomous vehicles. To this end, on-Vehicle Machine Learning (oVML) is a persuasive solution wherein each vehicle maintains their best machine learning model and is thereby capable of making intelligent decisions, even when it loses connectivity for some time. Training such an oVML model could require more samples of data than those per each vehicle. As a result, it demands data trading (and knowledge exchanges) with neighboring vehicles [4], [5]. In this paper, we address the challenge of training each oVML model by exploiting federated learning with their neighboring vehicles [6]–[10]. Without well-designed loyalty mechanism, vehicles will be unwilling to join federated learning tasks, which may hinder the adoption of federated learning. One may also adopt the contract theory approach along the lines of [11] to design an effective incentive mechanism for attracting vehicles with highly valuable data to participate in the learning.

Machine learning techniques in various fashions have already been applied to improve the performance of autonomous vehicles [12]: computer vision for analyzing obstacles, machine learning for adapting their pace to the environment (e.g., bumpiness of the road). Due to the potential high number of self-driving cars and the need for them to quickly respond to real world situations, traditional cloud approach generates safety risks. Federated learning can represent a solution for limiting volume of data transfer and accelerating such learning processes of autonomous vehicles. The anticipated outcome is a systematic approach for such design that transforms our existing vehicles into mobile data centers, performs federated learning and reacts timely to their needs. Resulting benefits include a better provision of seamless messages transfer and low delay internet services on the move for such connected autonomous vehicles.

Due to the potentially large number of anticipated autonomous cars and the need for them to quickly respond to real-world situations, the existing cloud-based learning approach is sluggish and may generate safety risks. One main problem is that locally collected data samples are owned by each oVML. Thus, their trading and knowledge sharing should keep the raw data private from other neighboring vehicles. In this regard, as proposed by Google's federated learning (GFL) [7], each oVML trades its locally trained model update (mainly, gradient parameters and learning weightage) rather than the raw data. Federated learning can represent a solution for limiting the volume of data transfer and accelerating

Manuscript received October 23, 2019; revised January 21, 2020 and March 18, 2020; accepted April 20, 2020. Date of publication April 27, 2020; date of current version August 14, 2020. The associate editor coordinating the review of this article and approving it for publication was M. Bennis. *(Corresponding author: Shiva Raj Pokhrel.)*

Shiva Raj Pokhrel is with the School of IT, Deakin University, Geelong, VIC 3220, Australia (e-mail: shiva.pokhrel@deakin.edu.au).

Jinho Choi is with the School of IT, Deakin University, Geelong, VIC 3220, Australia.

Color versions of one or more of the figures in this article are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TCOMM.2020.2990686

[1] An early version of this work has been accepted for presentation in IEEE WCNC Wksps 2020 [1].

the learning processes. Besides, it is worth noting that the regeneration of the raw data from the traded model is not possible, thus guaranteeing privacy. Such trading in GFL is handled by the aid of a centralized server that produces a global model, which is an ensemble average of all the locally trained model updates. Thereafter, each oVML downloads the globally updated model and computes their next local update until the completion of the global model training process. Observe that, because of the closed-loop exchanges (locally trained model update followed by a globally aggregated model update triggering next iteration of local training), the delay incurred in training completion of GFL may sometimes be around several minutes (10 or more), as reported recently for Google's keyboard application [13]. In oVML, in addition, a centralized server may not be available.

To this end, we propose (and evaluate) a blockchain-based federated learning (BFL) model, as illustrated in Figure 1, for efficient communication of autonomous vehicles as GFL is not straightforwardly applicable because of the following two fundamental shortcomings.

i) *Centralization:* GFL depends on a single global server, which is vulnerable to server's malfunctioning, is highly dependent on network connectivities and suffers heavily from bottleneck (as a result of the traffic of local model updates from each oVML). Such malfunctioning mostly incurs an inaccurate global model, increases the response time and creates distortion over oVML.

ii) *Loyalty:* An oVML with a large number of data samples contributes heavily to the global training. However, GFL lacks a mechanism to reward such oVML. Without a loyalty program, such a vehicle with highly useful data samples and information will not be motivated to federate evenly with the other vehicles with only a few samples.

To resolve the aforementioned two limitations of GFL, we leverage blockchain [14]–[20] (*remove the centralized global server of GFL and use a blockchain*) and propose a *blockchain-based federated learning* (BFL) approach, where the network system enables exchanging local model updates from vehicles while providing and verifying their corresponding rewards [21]. Figure 1 illustrates our proposed BFL, which suits to the autonomous vehicular network system by overcoming the centralized malfunctioning problem and extending the range of its federation to untrustworthy vehicles in a public network thanks to a validation process of the local training modules. More importantly, by providing rewards proportional to the usefulness of data sample sizes, BFL encourages vehicles with a larger size of data samples to join federated learning.

An abstract view of the proposed BFL framework consisting of autonomous vehicles and miners is shown in Figure 1. The miners can physically be either be moving vehicles or separate nodes at network edges (such as cellular base stations), which are relatively computationally powerful for the *mining process*.[2] We propose a uniform random vehicle-miner

[2]Mining in blockchain is a race between all the contending miners to generate (and validate) a block to append to the blockchain. To have a high likelihood of winning this mining race, it requires substantial computational resources, and the reward for doing so is the mining reward.

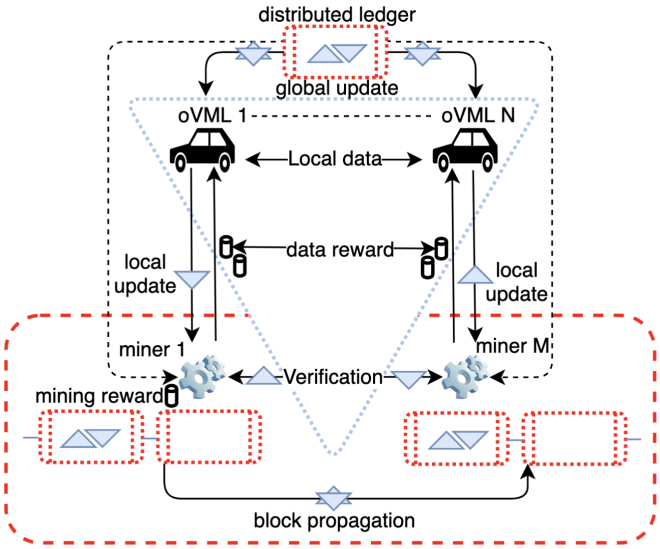


Fig. 1. Interaction between components of our blockchain-based federated learning approach for autonomous vehicles.

association scheme. Vehicle miner association using uniform random approach (as its source of randomness) in their association is automatically trusting either the miners, or the oVMLs of our BFL system; both of which are often anonymous entities to each other. The main benefits of random vehicle-miner association scheme are as follows:

i) an adversary who can compromise miners in other fixed association scheme cannot exploit this association mechanism to inform which oVMLs it aims to compromise.

ii) it defines a game on the blockchain that incentivizes its players to play randomly, relies on incentives, and does not make any assumption about their honesty.

iii) it is easier to form representative groups from the overall vehicles in the BFL system.

iv) it allows the global model and findings to apply to the entire BFL System.

v) it is simple, computationally cheap and reduces the risk of conscious or unconscious bias. The interactions in Figure 1 are explained as follows. Each oVML performs local learning and sends the local model update to its associated miner in the network. All miners exchange and verify all of their local model updates, and then execute their *proof-of-work* [22]. When a miner finishes its *proof-of-work*, it generates a block by recording the validated local model updates. In the end, the block (generated block consisting of the aggregate local model updates) is inserted into the *distributed ledger* of the blockchain. This newly inserted block can then be used by all associated vehicles to compute the required global model update.

With BFL, observe that the global model update is locally computable at each vehicle. Moreover, our BFL design also guarantees that the malfunctioning or failure of a miner, node or a vehicle does have no adverse impact on the global model updates for any other vehicles (overall network system). However, there is a tradeoff: in order to grasp all of the aforementioned benefits, as opposed to the GFL, BFL has an additional delay due to the blockchain management in

the system. To quantify (and address) the carried-over the delay by blockchain, we conduct the aggregate end-to-end delay analysis of the system with BFL by accounting the *proof-of-work*, communication and computation delays. With relevant insights from the model, we also minimize the overall perceived delay of the system by dynamically adjusting the block arrival rate, i.e., the complexity of *proof-of-work* for the blockchain system.

Based on the above discussion, the main contributions of this paper can be outlined as follows.

1. We propose and develop a novel and fully decentralized blockchain-based federated learning framework to ensure end-to-end trustworthiness and delay for the impending autonomous vehicular networking system.
2. We develop a comprehensive mathematical analysis of the system dynamics for the end-to-end delay analysis, which provides important insights into minimizing delay and deriving optimal block arrival rate by considering communication latency and consensus delays simultaneously.
3. By structuring an analogous vehicular network model, which incorporates the complex interactions of the wireless channel and link characteristics with the detailed transmission mechanism of the cellular networks, we succeed in measuring the impact on the performance of our BFL system. We perform extensive numerical and simulation experiments to evaluate the proposed framework under diverse network scenarios and channel conditions.

Due to the time-varying nature of our autonomous vehicular network settings, i.e. the number of vehicles, miners and wireless channel conditions vary dynamically in our network scenario. Therefore, we move a step forward to design an online algorithm for minimizing the system delay under realistic network conditions.

4. Based on the closed-form expression derived using the developed analytic model under static network conditions, we propose a real-time online algorithm that continuously monitors the deviation in the system delay from a targeted value and adapts the block arrival rate to ensure minimal aggregate delay and cope elegantly with the transient network dynamics.

More importantly, with relevant insight from this work, we find a number of challenges (gaps in knowledge) need to be resolved in order to realise these changes. In particular, we identify the following three key bottleneck challenges requiring further investigations, and provide potential future directions (Sec. VII):

- development of sophisticated mobility models considering specific features of autonomous vehicles for distributed learning;
- design of risk analysis and mitigation approach for privacy leakage during verification; and
- investigation of context-aware integrated (and robust) distributed learning based blockchain system for ultra reliable low delay applications.

II. Federated Learning: Problem and Solution

Consider $n = 1, 2, \ldots, N$ be the index of autonomous vehicles in a network system and let the n^{th} vehicle collects a set of data samples, s_n and computes its local learning update. The local learning update of the n^{th} vehicle is sent to the associated miner m that is randomly (uniformly) selected from a set $m \in \{1, 2, \ldots, M\}$. Here, N and M represent the numbers of vehicles and miners, respectively. Our federated training is a regression problem that focuses on solving the problem in parallel by considering the entire state space of data samples, $S = \cup_{n=1}^{N} s_n$ collected by all vehicles in our network system. For a global vector v, our objective is as follows:

$$\text{Minimize } \mathcal{F}(v), \tag{1}$$

where

$$\mathcal{F}(v) = \frac{\sum_{n=1}^{N} \sum_{d_j \in s_n} (a_j^{\intercal} v - b_j)^2 / 2}{|S|} \tag{2}$$

for convenience, $d_j = \{a_j, b_j\} \in S$ denote the j^{th} data sample, where a_j and b_j are the k-dimensional column vector and a scalar respectively.

With GFL, it is well-known that the default idea to solve Equation (1) is to perform local training at each oVML by using the stochastic gradient algorithm, followed by a global training for aggregating local updates via distributed Newton's method. In each stage, the oVML local model is recomputed with the number I iterations. Given the step-size $\delta > 0$, the local vector denoted by v_n is updated after every iteration as follows:

$$v_n^{t+1} = v_n^t - \frac{\delta}{I}\Big(\big(\nabla \mathcal{F}_j(v_n^t) - \nabla \mathcal{F}_j(v)\big) + \nabla \mathcal{F}(v) \Big),$$

where

$$\mathcal{F}_j(v) = \frac{(a_j^{\intercal} v - b_j)^2}{2}. \tag{3}$$

Every oVML in GFL sends $(v_n, \nabla \mathcal{F}_j(v))$, so called local update, to the centralized global server, which then computes the global update, i.e. $(v, \nabla \mathcal{F}(v))$. However, in our proposed GFL framework, the global server is replaced by a blockchain mechanism which is explained below.

A. Decentralized Solution With Blockchain

With BFL, we design a common trustworthy framework for sharing local model updates from all oVML of vehicles via a distributed ledger, where the blocks hold local updates and their validation is performed by using M miners (recall Figure 1). More specifically, each block in the ledger consists of two sections, namely *header* and *body*. The header of a block contains three important parameters: i) *pointer to previous block*, ii) *block arrival rate (denoted by Λ) in blocks per seconds*, and iii) *outcome of the proof-of-work*; and its body carries i) local computational delay and ii) the local update $(v_n, \nabla \mathcal{F}_j(v))$. Each miner maintains a contender block for dumping local updates from all the associated oVMLs and other contending miners. This dumping process carries on and halts either by timeout, τ_{out}, or once the block is

full (occupancy reaches maximum block size). Thereafter, the miner generates (randomly) the hash value iteratively by changing its inputs along the lines of that of *proof-of-work*, which terminates only after the final iteration guaranteeing that the generated size of the ultimate hash satisfies the desired target value. More importantly, the contender block is allowed to be a new block in the blockchain system only when the miner successfully generates the hash. This is immediately followed by the propagation of the newly generated block into the system, which acknowledges all other miners and mandates them to i) stop their *proof-of-work* computation and ii) add the new block to their ledgers. However, there is always a likelihood that in the mean time (before receiving the propagated block) another miner may also generates a block, causing updates in the ledger of its neighboring miners with the later generated block. Such unwanted consequences due to block propagation delay is know as *forking* in the blockchain system [23]. With BFL, forking creates a situation where some oVMLs unknowingly utilize wrong global update to their next iteration of local model update.

Based on the above discussion, it is worth noting that the block arrival rate denoted by Λ depends on the difficulty level of the corresponding *proof-of-work* mechanism, i.e., Λ is controllable by tuning the *proof-of-work* appositely. For example, the lower the desired block arrival rate $\Lambda^{\star}$ is, the smaller the desired target value for hash becomes. More importantly, we have now observed that the occurrence of forking depends mainly on two factors: i) block arrival rate (Λ), ii) block propagation delay, in particular, the probability of occurrence of forking increases with Λ and propagation delay, which is investigated in detail in Sections III and IV.

B. Loyalty Solution With Blockchain

The blockchain system also has the required loyalty program for promotion and rating of vehicles as expected. As a result, with BFL, our design not only provide rewards for data samples to the vehicles (*data reward*), but also supply rewards for the block verification process to the winning miners (*mining reward*). The flows of both data and mining rewards in the BFL system are illustrated as in Figure 1. The data reward to the vehicle is proportional to the size of local data samples from the oVML and is provided by its associated miner. The idea of the mining reward in BFL is as follows: *miners are paid rewards proportional to their contributed mining power*, similar to that in seminal blockchain system. The mining power of a miner is determined by the aggregation of the useful data samples of its associated oVMLs. There is room of adversary/selfish vehicles to inflate their data samples with arbitrary local model updates. However, miners can potentially verify trustworthy local updates by comparing the sample size with their local computation time. It is known that this can be formally assured in practice by using Intel's extensions, which has successfully been applied in blockchain system [24], [25].

Remark 1: We use the *proof-of-work* for tractability [16], [26], however, use of other consensus algorithms such as *Byzantine-fault tolerance* and *proof-of-stake* into BFL is quite straightforward, which may require more sophisticated computations and preambles to arrive at a consensus among miners. Further, *proof-of-work*'s strength lies in its ability to provide a mechanism for validity to a seemingly unlimited, unknown set of nodes which is important in our context. In addition, leading auto manufacturers such as General Motors and BMW has already started using blockchain as a way to share self-driving car data among themselves and other automakers.[3] Next, for oVML to be incentivized, *proof-of-work* is always accompanied by a reward structure that rewards the nodes for their work, as seen in Bitcoin, or Ethereum with the block rewards.

Algorithm 1 oVML Algorithm at Vehicle

procedure UPDATE LOCAL MODEL
  for $(i = 0; i \leq I; i++)$ **do**
vehicle n computes Equation (3)
  end for
end procedure
procedure UPLOAD LOCAL MODEL
associate vehicle n with miner m (uniform random)
vehicle n uploads local updates and local computation time
end procedure
procedure DOWNLOAD GLOBAL MODEL
vehicle n downloads new block from its miner m
end procedure
procedure UPDATE GLOBAL MODEL
  while $((v(t+1) - v(t))^2 > tol)$ **do**
compute global vector $v(t+1) \leftarrow v(t) + \sum_n \frac{I}{|S|}(v_n^t - v(t))$
  end while
end procedure

C. BFL Algorithms Design

Based on the above discussion, we develop two algorithms: i) oVML algorithm at the vehicle; and ii) algorithm at the miner as shown in Algorithm 1 and Algorithm 2 respectively.

Recall that GFL is highly vulnerable to the malfunctioning of centralized server which may cause distortion to the global models of all vehicles. In contrast, our proposed BFL computes global model update locally at each vehicle, which not only increases robustness against malfunctioning of the centralized server but also i) reduces the computational delay and ii) eliminates the global update propagation delay completely (from the centralized server to vehicles).

III. BFL Block Arrival Process

Tracking the block arrival process in BFL system can serve as a means for analyzing the performance of such a system. There are different approaches to quantify the block arrivals in blockchained systems, the two well-known schemes are: i) using the timestamp inside the block header ii) recording the time when the block arrives at the miners/oVMLs. The arrival time of a block at a miner depends on when the block

[3]https://www.coindesk.com/gm-bmw-back-blockchain-data-sharing-for-self-driving-cars.

Algorithm 2 Algorithm at Miner

procedure CROSS VERIFICATION
miner m broadcasts the obtained local model update
  while (block size $<$ max. size) or ($timer < \tau_{\text{out}}$)) **do**
    if (verify local update) **then**
Insert local update into contender block
block size$\leftarrow$ block size + size (local update)
    end if
  end while
end procedure
procedure GENERATE BLOCK
  while (hash satisfies target) or (receives generated block) **do** execute proof-of-work
    if (hash satisfies target) **then**
create new block(contender block)
      if (receives generated block) **then**
        if (forking) **then**
send acknowledgments.
        end if
      end if
    end if
  end while
end procedure
procedure PROPAGATE BLOCK
broadcast new block
  procedure AVOID FORKING
wait for acknowledgments (time out after τ_{out})
  end procedure
end procedure

is mined and how long it takes for propagation through the network. Once the block has been propagated in the network, then the information contained within the block is available for the miners and vehicles, and it is quite unlikely that any other miner will cause conflict by mining on the top of the generated block. On the other hand, the timestamp based scheme using the block header replies on the local clock of the miner (and may introduce error).

Without loss of generality, we consider block arrival process A in our BFL system as a Poisson point process on $\mathbb{R}$, with the following two attributes:

1) The process A has random number of points $A([a,b))$ located in a bounded interval $[a,b) \in \mathbb{R}$, which can be represented by a Poisson random variable with mean $\lambda([a,b))$ as a Radon measure (non-negative).
2) The points of the process A located in intervals $[a_1,b_1),[a_2,b_2),\ldots,[a_k,b_k)$ forming k independent random variables with expectations $\lambda([a_1,b_1)),\ldots,\lambda([a_k,b_k))$.

For a point process $\{A_{(i)}\}_{i\geq 1}$ defined on nonnegative reals, we can arrange the ascending order of the points based on their arrival times, $A_1 \leq A_2 \leq A_3 \leq \ldots$. Therefore, the effective distance between two points is a random variable, $\mathcal{T}_i = A_i - A_{i-1}$, for $\mathcal{T}_1 = A_1$ and $i = 2,3,4,\ldots$, so called *inter-arrival times or arrival delay* in our BFL system. Now we have the following two different cases [27].

CASE I (Homogeneous Poisson Process): For $\{A_{(i)}\}_{i\geq 1}$ being a homogeneous point process with rate Λ, the corresponding arrival delays $\mathcal{T}_i$ are independent and identically distributed (iid) exponential random variables with average $1/\Lambda$, therefor, by using memoryless property:

$$\mathbb{P}(\mathcal{T}_i \leq t) = 1 - e^{-\Lambda t}. \tag{4}$$

CASE II (Nonhomogeneous Poisson Process): For $\{A_{(i)}\}_{i\geq 1}$ being a nonhomogeneous point process with intensity $\Lambda(t)$, the corresponding arrival delays $\mathcal{T}_i$ can be estimated as follows. The first arrival time $\mathcal{T}_1 = A_1$ has the following distribution

$$\mathbb{P}(\mathcal{T}_1 \leq t_1) = 1 - e^{-\int_0^{t_1} \Lambda(x)dx}. \tag{5}$$

Thereafter, given the first arrival delay $\mathcal{T}_1 = t_1$, the (conditional) distribution of second arrival time $\mathcal{T}_2$ is

$$\mathbb{P}(\mathcal{T}_2 \leq t_2 | \mathcal{T}_1 \leq t_1) = 1 - e^{-\int_{t_1}^{t_2} \Lambda(x)dx} \tag{6}$$

and so on for $i = 3,4,\ldots$. For a nonhomogeneous Poisson process $\{A_{(i)}\}_{i\geq 1}$, each point is independently distributed on the interval $a \in [0,t]$ with the following distribution:

$$\mathbb{P}(A_i \geq a) = \frac{\lambda(a)}{\lambda(t)}, \tag{7}$$

$$\text{where } \lambda(t) = \lambda([o,t)) = \int_0^t \Lambda(t)dt \tag{8}$$

It is known that when the distribution of A_i is invertible, each A_i can be transformed into a uniform random variable on $[0,1]$, resulting in a number of independent uniform random variables. In fact, $\lambda(t)$ transforms a Poisson process into a homogeneous Poisson process and consequently, statistical methods for nonhomogeneous Poisson process often transforms the data for their performance analysis [28].

IV. TOWARDS OPTIMAL BFL SYSTEM

In this section, our objective is to minimize the total system delay, T_{tot}, by investigating its dependency on the network and BFL parameters such as transmission frame size, block size, block arrival rate etc. The expectation of T_{tot}, denoted by $\mathbb{E}_{n,m}[T]$ when n vehicles and m miners actively participating in the system, is determined by aggregating i) the mining consensus delay for block verification and generation, and ii) the communication delay of the overall system. The overall picture of different submodels and their interactions developed for performance analysis is as illustrated in Figure 2.

We consider four assumptions to make our analysis tractable.

Assumption 0: The data collected by the oVMLs are unbalanced and non-IIDs.

The IID sampling of the training data is important to ensure that the stochastic gradient is an unbiased estimate of the full gradient [29]. However, it is unrealistic to assume that the local data on each oVML is always IID.[4] As the data present on

[4]McMahan *et al.* [30] have demonstrated that their FL work with certain non-IID data.

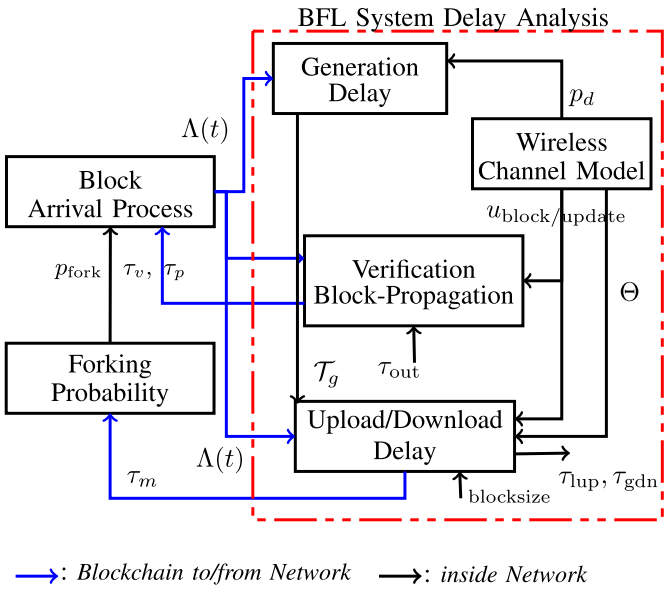


Fig. 2. Interactions that couple the submodels are shown. The submodels within the dashed rectangle (representing BFL system delay dynamics) take the arrival rate $\Lambda(t)$ as input, and provide different delays as outputs. The two submodels outside the dashed rectangle (representing Blockchain dynamics) estimate arrival rates and forking probabilities.

the individual oVMLs is collected by the vehicles themselves based on their local environment and pattern, both the size and the distribution of the datasets typically vary between different oVMLs.

Assumption 1: The computation and verification duration inside oVML is sufficiently small as compared to communication delay in our proposed BFL system.

As computationally powerful servers are installed inside the vehicles, it is reasonable to assume that the wireless communication channel is the bottleneck for the proposed BFL system, and the local and global learning times inside vehicles are typically negligible.

Assumption 2: The *proof-of-work* follows a Poisson process.

Note that Assumption 2 is a standard assumption that have been used in [20], [31]–[34]. Also recall our explanation of block arrival process in Sec. III. Gobel *et al.* [31] and Rosenfeld [32] pointed out that the blocks arrive as a Poisson process and the number of blocks generated is a random variable with negative binomial distribution. Rosenfeld [33] also assumed Poisson process. Nakamoto *et al.* [34] performed their calculations using a similar Poisson process assumption. Bowden *et al.* [20] has examined this assumption and demonstrated that, at certain timescales, the block arrival process is not Poisson, however they proposed a refined point process which can be transformed into a Poisson process (see [20, Sec. IV]). We therefore make Assumption 2.

Assumption 3: All miners are synchronized with each other and start their *proof-of-work* at once by maintaining $timeouts$.

Assumptions 1, 2 and 3 facilitate quantitative analysis, while hardly affecting the accuracy of the results. We quantify the various delays of the system as illustrated in Figure 2 using a pragmatic approach in the following subsections.

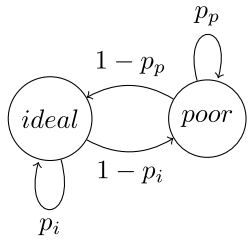


Fig. 3. Approximate Cellular Channel State Transition Diagram.

A. Modeling Wireless Channel

The communication delays are entailed by the global and local data model transport between oVMLs and miners. This is denoted by the two different procedures in Algorithm 1, namely, upload local model (Steps 6-9) and download global model (Steps 10-12). The analysis of communication delay to estimate the uploading, downloading, verification and block propagation delay in our proposed blockchain based system requires an accurate analysis of the underlying Cellular (e.g., V2V and V2X LTE) channel dynamics. In the following, we develop approximate analysis considering channel state dynamics of [35] and include only the essential details.

We assume discretized time-slots, where the duration of each slot is the transmission time for a block or module in our system. As illustrated in Figure 3, we assume that the underlying channel dynamics changes between an ideal state (in which all transmissions are successful) and a poor state (in which all transmissions are failed). Under this setting, observe that there are only two possible outcomes for each transmission, often designated success or failure, therefore, the probability of successful transmission, is the same for every trial and follows geometric distribution.

Let Θ denote the maximum rate at which the channel can transmit the link layer frames to the vehicle, (i.e., Θ =*bit rate÷Frame size in bits*). Let ν be the speed of the vehicle, and f_c the carrier frequency, the Doppler frequency is $f_d = f_c\nu/c$ ($c = 3 \times 10^8$ m/s). Considering F as the fading margin and given the physical modulation and coding schemes, the channel is in the poor state, if received Signal to Noise Ratio (SNR) is below a threshold $\mathbb{E}[\text{SNR}]/F$,; else, it is in the ideal state. The average probability that a frame transmission fails due to errors in the channel is $\bar{p}_e = 1 - e^{-1/F}$. Representing $\eta = \sqrt{2/(F(1-\rho^2))}$, $\rho = J_0(2\pi f_d/\Theta)$, with, ρ is the Gaussian correlation coefficient of two samples of the amplitude of a fading channel with frequency f_d; $1/\Theta$ is the frame transmission time over the channel; and $J_0(.)$ is the zero order Bessel function. The stationary state transition probabilities (recall Figure 3) with the Marcum-function $\mathbb{Q}(.,.)$ are as follows:

$$p_p = \boldsymbol{Pr}(\text{Remains in poor}) = 1 - \frac{\mathbb{Q}(\eta, \rho\eta) - \mathbb{Q}(\rho\eta, \eta)}{e^{\frac{1}{F}} - 1},$$

$$p_i = \boldsymbol{Pr}(\text{Remains in ideal}) = \frac{1 - \bar{p}_e(2 - p_p)}{1 - \bar{p}_e}.$$

Let L be the required number of link layer frames for transmitting a block or a local update in the cellular channel, then (for unbalanced and non-IID data sets collected by

the vehicles, recall ASSUMPTION 0, although we omit the vehicle-indexing in the equations for simplicity)

$$L_{\text{block/update}} = \frac{\text{Block/Local update size in bits}}{\text{Frame size in bits}}.$$

With the Markovian process embedded at the starts of frame transmissions, we denote ℓ_i as the probability that at least one frame fails out of i link layer transmissions, when the channel state is $ideal$; and $\nu_i^{(r)}$ the probability that at least one fails out of i link layer transmissions, given that the first link layer frame has already had r (with $r < R$, here, R is the maximum number of (re)transmissions) transmission attempts failures while the channel state is at $poor$. The renewal equations to estimate these probabilities are

$$\ell_i = p_i \ell_{i-1} + (1-p_i)\nu_{i-1}^{(0)}$$
$$\nu_i^{(r)} = (1-p_p)\ell_i + p_p \nu_i^{(r+1)}.$$

Here, ℓ_i, $1 \leq i \leq L$, and $\nu_i^{(r)}$, $0 \leq r \leq R$, are obtained by imposing the conditions

$$\ell_1 = 0, \ \nu_1^{(r)} = {p_p}^{(R-r)}, \ \text{and} \ \nu_i^{(R)} = 1.$$

A block (or an update) is discarded in the channel with the following probability:

$$p_d = \nu_L^{(0)} \frac{(1-p_i)}{(2-p_p-p_i)} + \frac{\ell_L}{1+\frac{(1-p_i)}{(1-p_p)}}. \tag{9}$$

The effective block/update rate over the channel, accounting the link layer (re)transmissions, can be computed with the following set of renewal equations:

$$u_i = 1 + p_i u_{i-1} + (1-p_i)\varrho_{i-1}^{(0)}$$
$$\varrho_i^{(r)} = 1 + p_p \varrho_i^{(r+1)} + (1-p_p)u_i$$
$$\varrho_i^{(R)} = 1 + p_p \varrho_{i-1}^{(0)} + (1-p_p)u_{i-1}$$

with boundary conditions $u_1 = 1$ and $\varrho_1^{(R)} = 1$. Here, i) given that the initial channel state is $ideal$, u_i is the average link layer frames needed to be transmitted for a block/update consisting of i link frames; ii) $\varrho_i^{(r)}$ is the average number of link layer frames needed to be transmitted for a block/update given that the first link layer frame has already had r failed transmission attempts and the initial state is $poor$.

Finally, the expected number of frames needed to be transmitted for a block/update consisting of L frames is

$$u_{\text{block/update}} = \frac{u_L(1-p_p)}{(2-p_p-p_i)} + \frac{\varrho_L^{(0)}}{1+\frac{(1-p_p)}{(1-p_i)}}. \tag{10}$$

Observe that (10) is straightforward applicable for unbalanced and non-IID data sets collected by the vehicles, recall ASSUMPTION 0, even though we omit the vehicle-indexing in the (10) for simplicity. Based on the above analyses, with slight abuse of notation and using our proposed framework (see Figure 1), *the communication delay* experienced by the vehicles while uploading the local updates and downloading global model (i.e., a block from the distributed blockchain ledger) can be estimated as (applicable for unbalanced and non-IID data sets collected by the vehicles as well, recall ASSUMPTION 0, although we omit the vehicle-indexing for simplicity)

$$\tau_{\text{lup}} = \frac{u_{\text{update}}}{\Theta} \ \text{and} \ \tau_{\text{gdn}} = \frac{u_{\text{block}}}{\Theta} \tag{11}$$

respectively.

Remark 2 (With Mobility Induced Handoff): Our analysis can be extended to handle mobility induced handoffs. For the case of loss and delay accounting handoffs, the computation of p_d (in Eqn. (9)) and the communication delays (in Eqn. (11)) can be modified as

$$\hat{p}_d = p_h + (1-p_h)\left[\nu_L^{(0)} \frac{(1-p_i)}{(2-p_p-p_i)} + \frac{\ell_L}{1+\frac{(1-p_i)}{(1-p_p)}}\right],$$
$$\hat{\tau}_{\text{lup}} = \frac{u_{\text{update}}}{\Theta(1-p_h)} \ \text{and}$$
$$\hat{\tau}_{\text{gdn}} = \frac{u_{\text{block}}}{\Theta(1-p_h)} \tag{12}$$

where p_h is the likelihood that a block (or an update) is discarded in the channel due to mobility induced handoffs.

B. Modeling Blockchain Delay

As illustrated in Algorithm 2, the blockchain delay consists of block arrival, propagation and verification delays. We assume that the verification time is sufficiently small (almost negligible) when compared to the communication delays. Therefore, considering m contending miners the total cross verification delay (Steps 1-9, Algorithm 2), denoted by τ_v, of the system can be found as

$$\tau_v = \max\left\{(\tau_{\text{out}} - \tau_{\text{lup}}), \sum_m \frac{u_{m_{\text{update}}}}{\Theta}\right\}, \tag{13}$$

where $u_{m-\text{update}}$ is the expected number of frames needed to be transmitted from miner m. Denote by $\mathbb{E}[SNR_m]$ the perceived expected SNR by miner m while receiving frames from other miners, which is required for computing u_m-*update* (along the same lines of (5)).

Similarly, the total block propagation delay, denoted by τ_p, from the winning miner $\hat{m}$ (Step 22, Algorithm 2) is given by

$$\tau_p = \max\left\{\tau_{\text{out}}, \sum_m \frac{u_{\hat{m}-\text{block}}}{\Theta}\right\}, \tag{14}$$

where $u_{\hat{m}-\text{block}}$ is the expected number of frames needed to be transmitted from the miner $\hat{m}$ (and can be computed along the lines of (5)).

Finally, the block arrival delay (Steps 10-21, Algorithm 2), denoted by $\mathcal{T}_g$, can be estimated as follows. Using Assumption 2, the block arrival delay can be modeled by a random variable with an exponential distribution (Sec. III). By fitting the distribution of block arrival to the exponential distribution, the mean or expected value of such an exponentially distributed random variable (i.e., the block arrival delay) with block arrival rate parameter Λ is given by (using (4) and (9))

$$\mathcal{T}_g = \frac{1}{(1-p_d)\Lambda}. \tag{15}$$

Observe that, using (9), the delay of interest in this case is the block arrival delay of the winning miner, given that the block is not discarded with probability p_d in the channel.

Finally, we compute the expected overall system delay (*communication and computing delays*), denoted by $\mathbb{E}[T]$ by, using a renewal reward approach. With n vehicles and m miners, the expectation $\mathbb{E}_{(n,m)}[T]$ can be computed as

$$\mathbb{E}_{(n,m)}[T] = \big(\tau_{\rm local} + \tau_{\rm lup} + \tau_v + \tau_p + \mathcal{T}_g + \tau_{\rm gdn} + \tau_{\rm global}\big) + p_{\rm fork}\big(\mathbb{E}_{(n,m)}[T] - \tau_{\rm gdn} - \tau_{\rm global}\big), \quad (16)$$

where $\tau_{\rm local}$ and $\tau_{\rm global}$ are the times involved in the local (Steps 1-4, Algorithm 1) and global (Steps 13-17, Algorithm 1) updates of the models, respectively. The forking probability $p_{\rm fork}$ required in (10) can be computed as

$$p_{\rm fork} = 1 - \prod_{m \neq \hat{m}} \boldsymbol{Pr}\big((\tau_m - \tau_{\hat{m}}) > \tau_p\big), \quad (17)$$

where $\tau_m = (\tau_{\rm local} + \tau_{\rm lup} + \tau_v + \mathcal{T}_g)$ is the time delay before miner m generates a block and $\tau_{\hat{m}}$ is that of the winning miner.

C. Minimizing Delay for Autonomous Vehicles

With relevant insights from the derived expected delay of the system (see (8)), we propose an approach to evaluate the optimal $\Lambda^\star$ that minimizes the delay of the involved *proof-of-work* process. Observe that the *proof-of-work* process affects i) block arrival/generation delay ($\mathcal{T}_g$), ii) block propagation delay (τ_p), and iii) forking probability ($p_{\rm fork}$), all of which are interlinked with each other due to the winning miner. Next, we have the following approximation.

Based on Assumption 3, all miners are synchronised with each other and start their *proof-of-work* process at once by maintaining $\tau_{\rm out}$ such that $\tau_v = \tau_{\rm out} - (\tau_{local} + \tau_{lup})$.

Assumption 3 also mandates that all miners wait till $\tau_{\rm out}$, even after completing the verification (Steps 1-9, Algorithm 2), which provides the following approximation:

$$\mathbb{E}_{(n,m)}[T] \approx \hat{T}_{n,m} = \frac{\big(\tau_{\rm out} + \mathbb{E}[\mathcal{T}_g]\big)}{1 - p_{\rm fork}} + \tau_{\rm global} + \tau_{\rm gdn}. \quad (18)$$

Therefore, our relaxed optimization is as follows:

$$\begin{aligned} &\text{Minimize } \hat{T}_{(n,m)} \quad (19)\\ &s.t.\ \forall\, n = \{2,3,\ldots\}\\ &\quad \forall\, m = \{2,3,\ldots\}\\ &\quad 1 > p_{\rm fork} \geq 0\\ &\quad \tau_{\rm out}, \tau_{\rm global}, \tau_{\rm gdn} > 0. \end{aligned}$$

Observe that the delays, $\tau_{\rm global}, \tau_{\rm gdn}$ and $\tau_{\rm out}$ in (17) are constant. Using Assumptions 2 and 3 for $p_{\rm fork}$ required in (17), $(\tau_m - \tau_{\hat{m}})$ becomes $(\mathcal{T}_g - \mathcal{T}_{\hat{g}})$. Thus, taking $\mathcal{T}_{\hat{g}}$ as the block arrival delay of the winning miner and by using the complementary cumulative distribution of the exponentially distributed $\mathcal{T}_g$, the complementary cumulative distribution of $\mathcal{T}_{\hat{g}}$ is given by

$$\boldsymbol{Pr}\Big(\mathcal{T}_{\hat{g}} > X\Big) = \prod_m \boldsymbol{Pr}\Big(\mathcal{T}_g > X\Big) = e^{-M\Lambda(1-p_d)X}. \quad (20)$$

Therefore, (16) simplifies to

$$p_{\rm fork} = 1 - e^{-M\Lambda(1-p_d)\tau_p}, \quad (21)$$

where τ_p is a constant under stationary network conditions.

Algorithm 3 Online Delay Minimizing Algorithm

1: **procedure** INITIALIZE
2: **if** $(n \neq n^{\rm new} \,||\, m \neq m^{\rm new})$ **then**
3: $k := 0$; $p_d^0 := 0$; $n := n^{\rm new}$; $m = m^{\rm new}$;
4: $\Lambda^0 = \frac{-m\tau_p + \sqrt{m^2\tau_p^2 + 4m^2\tau_{\rm out}\tau_p}}{2m^2\tau_p^2\tau_{out}}$;
5: Compute $\hat{T}^0_{n,m}(\Lambda^0, p_d^0, m)$ and $p^0_{\rm fork}(\Lambda^0, p_d^0, m)$ using (17) and (19).
6: **end if**
7: **end procedure**
8: **procedure** UPDATE
9: **while** $((\hat{T}^k_{n,m} - \hat{T}^0_{n,m})^2 > tolerance,\ \epsilon)$ **do**
10: Compute $\frac{d\Lambda^k}{d\hat{T}^k_{n,m}}|_{\hat{T}^k_{n,m} = \hat{T}^0_{n,m}}$ using (25)
11: $\Lambda^{k+1} \leftarrow \Lambda^k - \frac{1}{k}\Big(\frac{d\Lambda_k}{d\hat{T}^k_{n,m}}|_{\hat{T}^k_{n,m} = \hat{T}^0_{n,m}}\Big)(\hat{T}^k_{n,m} - \hat{T}^0_{n,m})$;
12: Compute p_d^{k+1};
13: $k \leftarrow k + 1$;
14: **end while**
15: **end procedure**

Then, the required expectation, $\mathbb{E}[\mathcal{T}_g]$ for (14) is

$$\mathbb{E}[\mathcal{T}_g] = \frac{1}{M\Lambda(1-p_d)}. \quad (22)$$

Using (19) and (21), (17) can be approximated by

$$\hat{T}_{n,m} \approx \big(\tau_{\rm out} + \frac{1}{M\Lambda(1-p_d)}\big)e^{M\Lambda(1-p_d)\tau_p} + \tau_{\rm global} + \tau_{\rm gdn}. \quad (23)$$

1) Approximate Offline Idea: Under perfect channel conditions, considering zero discards, (22) is a convex function of Λ, denoted by $f(\Lambda)$, and the required $\Lambda^\star$ (solution of (18)) can be computed as follows. Since the derivative of $f(\Lambda)$ with respect to Λ is

$$f'(\Lambda) = \frac{e^{\tau_p M\Lambda}}{M\Lambda^2}\Big(\tau_p\tau_{\rm out}M^2\Lambda^2 + \tau_p M\Lambda - 1\Big),$$

by setting it to 0, we have

$$\Lambda^\star = \frac{-M\tau_p + \sqrt{M^2\tau_p^2 + 4M^2\tau_{\rm out}\tau_p}}{2M^2\tau_p^2\tau_{out}}, \quad (24)$$

which minimizes $f(\Lambda)$ under perfect channel conditions ($1 - p_d \approx 1$).

2) Online Delay Minimizing Algorithm: With relevant insights from the analysis above, viz. (17), (21), (22) and (23), we develop an adaptive real-time algorithm for minimizing delay; see Algorithm 3 above. Note that the only way to minimize (17) is by minimizing $p_{\rm fork}$ and $\mathbb{E}[\mathcal{T}_g]$. However, given p_d and number of miners (denoted by m in the algorithm), the only way to minimize $\mathbb{E}[\mathcal{T}_g]$ is by increasing Λ (see (21)). Therefore, the fluid dynamical model for the Algorithm 3 can be written as

$$\frac{d\Lambda(t)}{dt} = \frac{1}{\alpha(t)}(\hat{T}_{n,m}(t) - \hat{T}^0_{n,m})^+_{\Lambda(t)} \quad (25)$$

where $\alpha(t) = -\beta_t^{-1}\frac{d\Lambda(t))}{d\hat{T}_{n,m}(t)}$ with the step-size parameter β_t decreasing as $1/t$. The notation $g(x)_y^+$ in (24) has the following meaning

$$g(x)_y^+ = \begin{cases} g(x), & y > 0 \\ \max(g(x), 0), & y = 0. \end{cases}$$

Observe in Step 11 of Algorithm 3 that $\beta_t = \frac{1}{k}$, where k is the iteration index used in the algorithm. Using (22), we compute (Step 10 of Algorithm 3),

$$\frac{d\Lambda^k}{d\hat{T}^k_{n,m}}|_{\hat{T}^k_{n,m}=\hat{T}^0_{n,m}} = \frac{m(\Lambda^k(1-p_d^k))^2}{e^{\tau_p m\Lambda(1-p_d^k)}}\Big(\tau_p\tau_{\text{out}}m^2(\Lambda^k(1 - p_d^k))^2 + \tau_p m\Lambda^k(1-p_d^k) - 1\Big)^{-1}. \quad (26)$$

The iteration index k resets to zero if there is a change in number of miners or vehicles, i.e., either $n(t) \neq n(t)^{\text{new}}$ or $m(t) \neq m(t)^{\text{new}}$ (Step 3 of Algorithm 3). Also, k increases by 1, when there is a significant increase in the delay (Step 13 and 9 of Algorithm 3).

3) On Convergence of Unbalanced and Non-IID Data: Observe that FL enables a large number of vehicles to jointly perform learning without sharing data privately. As a celebrated algorithm in this setting, FL averaging mechanism executes stochastic gradient descent in parallel on all subsets of the participating vehicles and computes the ensemble mean of the sequences only once in a while. Despite this simplicity, it often lacks theoretical guarantees under realistic settings. However, the convergence of FL averaging for non-iid data setting has recently been established [36].

Balanced data is taken for simplicity (as an example) in this paper. But, as noted earlier, any non-IID data based FL for which the function $f(\Lambda)$ is known can be considered. More importantly, we can adopt their [36], [37] results and bounds, which also demonstrates the tradeoff between communication efficiency and convergence rate. Exploiting this idea, as vehicles often may be disconnected from the server, we can always relax the assumption of full vehicle participation to partial vehicle participation and investigate the implications under different averaging schemes. Furthermore, using important insights from [36], the desired degree of vehicle participation rate may also be achieved without severely slowing down the learning, which requires further investigations.

V. Performance Evaluation

In this section, we evaluate the overall performance of the proposed system and gain important insights on the overall learning completion time for the system under different channel conditions, by conducting numerical and simulation experiments. The network setting and parameters used for our evaluation are based on 3GPP LTE Cat. M1 specification.[5]

Figure 4 illustrates the impact on the overall system delay with respect to block arrival rate, i.e. demonstrates a snapshot of $\mathbb{E}_{n,m}[T]$ vs. Λ (recall the closed form approximation derived in Equation (22)). As expected, the delay decreases

[5]see technical reports at https://www.3gpp.org/DynaReport.

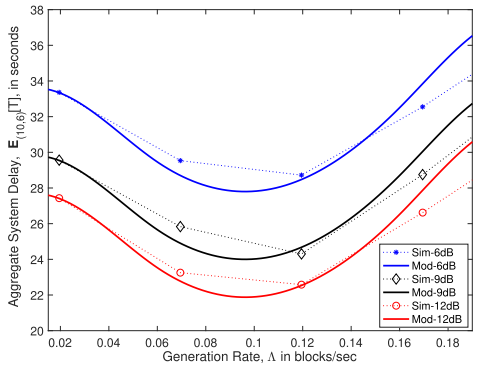


Fig. 4. Mean system delay with increase in block arrival delay under different channel conditions.

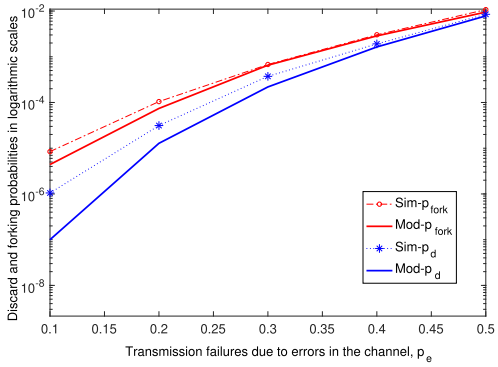


Fig. 5. Forking and discard probabilities under different channel conditions.

with increase in SNR, which is captured well by both numerical and simulation results. Furthermore, Figure 4 demonstrates that, for given network setting (in our experiment the attainable value of Λ under the given network condition of Figure 4 is $0.02 < \Lambda < 0.2$), the function, $f(\Lambda)$, is convex and its epigraph (the set of points on or above the graph of $f(\Lambda)$) is a convex set. In fact, it is known that if a twice-differentiable function of a single variable is convex, then its second derivative must be non-negative on its entire domain. This provides us with an important observation as follows:

$\mathbf{O}_1$) The minimization of $\mathbb{E}_{n,m}[T]$ can be achieved by adjusting Λ appropriately as the achieved local minimum of such a convex function is also a global minimum.[6]

Next, we evaluate the impact of wireless transmission failures on performance due to fading. Figure 5 depicts the probability that a block or a local update is discarded (due to repeated failures) in the channel. It also depicts the probability of forking with increase in transmission failures due to fading. It can be observed that both p_d and p_{fork} increase approximately log linearly, i.e., increase exponentially with the increase in transmission failures in the channel $\bar{p}_e$; p_d is slightly lower than p_{fork}. Our analytic modelling accurately captures these facts.

Since the increase in wireless transmission failures (increases discards, recall Equation (9)), on average, requires a higher number of retransmissions, before successful

[6]It is worth noting that a strictly convex function will have at most one global minimum however precise determination such strict convexity without Approximation 1 requires further investigations and is left for future work.

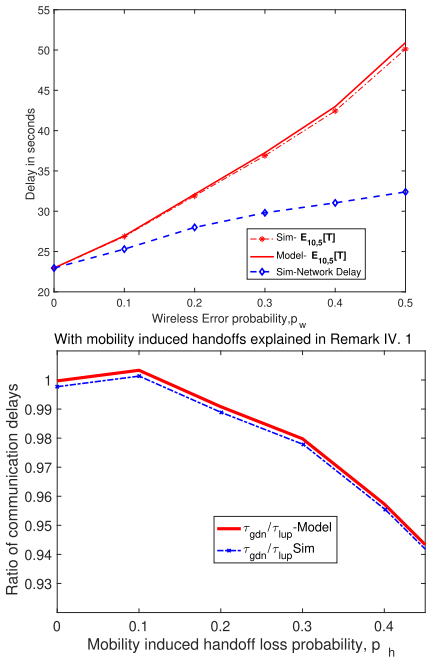


Fig. 6. Impact of channel errors and handoffs on the overall system and network delay.

transmission, which increases the service time of a frame in the channel. The increase in delay provides more time for contending miners to complete their block generation process (recall Equations (16) and (19)). Hence, our second main observation is as follows:

$\mathbf{O}_2$) With increasing transmission failures, *the increase on forking probability is higher than on* p_d, which adversely effects performance and causes substantial increase in the overall delay of our BFL system.

With a low probability of transmission failures (say $\bar{p}_e \leq 0.1$), the small loss is masked by the R (re)transmissions in that the discard probabilities are almost zero. With significant wireless channel errors, however, the $K = 7$ retransmissions are not enough to mask the repeated failures, and blocks are lost due to frame discards, which again leads to increase in the forking probability (Figure 5).

Under the same system settings of Figure 5, the overall system delay perceived by the vehicles and miners are shown in Figure 6. We observe that, with increase in wireless channel errors the block/update loss probability increases and the total throughput of the system decreases as expected and therefore increases the system delay. Of particular importance, our third main finding is as follows:

$\mathbf{O}_3$) Due to increase in forking probability, *the delay perceived by the miners and vehicles in the system is higher than that observed at the network level* in the presence of wireless losses. This new type of effect that we discover with BFL is in an opposite sense to the well-known phenomenon in GFL. Left panel of Figure 6 validates this observation with simulations for 10 vehicles and 5 miners and $\bar{p}_e \in [0, 0.5]$. Right panel of Figure 6 validates this observation with the same settings considering handoff losses $\bar{p}_h \in [0, 0.5]$. To tackle this unwanted phenomenon under time varying

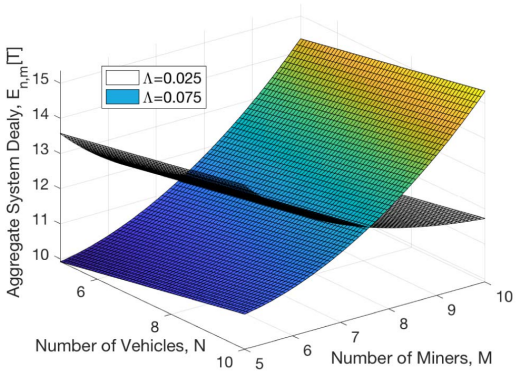


Fig. 7. Mean system delay with variation in number of miners and vehicles for two different block arrival rates (0.025 and 0.075 blocks/sec) under static channel conditions.

channel conditions and mobility induced handoffs, we investigate ways to minimize the delay by tuning the block arrival rate in the system appropriately (see Section IV C).

We conduct extensive numerical experiments to analyse the impact of variation in number of vehicles and miners on the overall learning delay of the system. Figure 7 illustrates the impacts with two different block arrival rate, $\Lambda \in \{0.025, 0.075\}$, under static channel conditions $\bar{p}_e \approx 0.2$. Given the block arrival rate, observe in Figure 7 that the delay is predictable with increase/decrease in the number of vehicles and miners. This observation provides us an important insight as follows:

$\mathbf{O}_4$) Λ *has the potential to be an adjustable tuning knob and can drive the overall system dynamics to the desired point*. This insight motivates us to minimize $\mathbb{E}_{(n,m)}[T]$ and find $\Lambda^\star$ under static and good (significantly small p_d) network conditions.

Remark 3: Note that our approach to minimize $\mathbb{E}_{(n,m)}[T]$ and find $\Lambda^\star$ assumes time-invariant settings. Also, ($\bar{p}_e \approx 0.2$) implies ($p_d \approx 0.000012$). Therefore, we propose Algorithm 3. Our BFL system and Algorithm 3 provides anticipated performance indication as long as the rate of variation in the BFL network setting is slower than the rate at which our approach converges and finds $\Lambda^\star$. The situation in which the BFL network environment evolves faster requires additional research towards jointly tuning network system parameters along with BFL parameters, such as retransmission limit, size of frames, block size and local updates. Furthermore, extensive investigations is required particularly for situations when fading or shadowing plays an important role in the channel.

With extensive numerical experiments, using Equations (22) and (23), we obtain the results for optimal block arrival rate, Λ^* with respect to the variation in number of miners and block propagation delay. Figure 8 illustrates two different views of the same result and explain the monotonic dependency of $\Lambda^\star$ with the variation in number of miners and propagation delay, which happens quite often in realistic vehicular networks. Figure 8 further provides a new finding as follows:

$\mathbf{O}_5$) Under the same BFL network settings, if there is need to increase (resp. decrease) block arrival rate, $\Lambda^\star$, then it is attainable by decreasing (resp. increasing) the halting time for

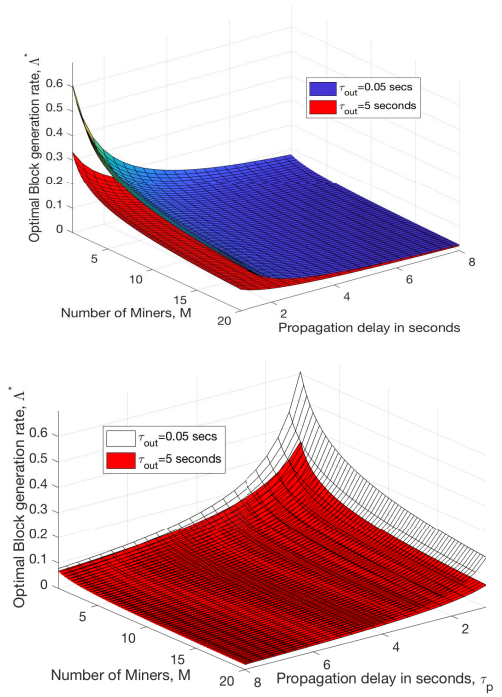


Fig. 8. Numerical results for the optimal block arrival rate, Λ^*. Observe the monotonic dependency of $\Lambda^\star$ with the number of miners and block propagation delays in the channel. Both left and right panels illustrate different views of the same result.

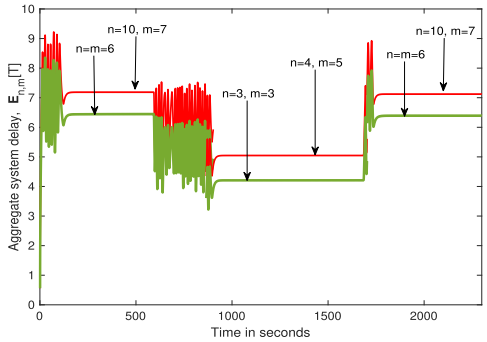


Fig. 9. Numerical results using Algorithm 3. Comparison of evolution of delays (in seconds) of BFL plus Algorithm 3 as observed in numerical experiments. Observe the perceived delays with the two different runs where the number of miners and vehicles changes differently.

the dumping process either through timeout, τ_{out} or via the block size parameter of our BFL framework. This observation requires further investigations.

To check the robustness of our Algorithm 3 to delay changes, we have performed extensive numerical experiments with $\epsilon = 3$ and $\bar{p}_e = Unif(0.2, 0.5)$.[7] We did this by changing the number of miners and vehicles. By changing the number we guarantee that there will be fluctuations in the system, thus creating delay variations. Numerical results in Fig. 9 confirm that our algorithm is robust against such delay changes. It is also interesting to observe that with the injected variations the delays perceived oscillates sometime before it converges may be due to random $\bar{p}_e$ or choice of ϵ, which needs further investigations and is left for future work.

[7]The function $Unif(a, b)$ returns a continuous value with in a given interval (a,b) with equal probability.

VI. Related Works

Yan *et al.* [38] summarised the state of the art of IoT trust management mechanisms. Other literature on distributed methods to establish trust is also quite rich (see [39] and references therein). These methods mainly proposed aggregation with multiple observer nodes. In particular, IoT nodes are corroborating with their nearby observers/observations independently, often by exploiting their spatial or temporal proximity while doing so. However, the distributed methods are not coupled tightly with the blockchains' trustworthy, transparency, and auditability mechanisms. There are few more recent works which focus on the integration and decentralization of the trust management and reputation management mechanisms using blockchain-based IoT applications [16], [19], [40], [41]. However, the interactions between the IoT nodes and trust management are mostly application and network specific.

Understanding and improving the working mechanism of blockchain and federated-learning has become more and more important in vehicular networks due to the anticipated growth of connected autonomous vehicles with diverse quality of service requirements. Most of the existing work ([7]–[9], [12], [14]) that explains and quantifies the performance of blockchain (resp. FL) over wireless focuses on the blockchain part (resp. distributed learning) and does not consider the impact due to the interaction with the packet transmission mechanism in the lower layers of Cellular networks.

VII. Future Challenges

This research work is motivated to invent *a systematic approach for distributed learning design that transforms our vehicles into mobile data centers, performs federated learning and reacts timely to their needs*. Resulting benefits should include a better provision of seamless messages transfer and low delay internet services on the move for connected autonomous vehicles. In this context, with relevant insights from this work, we have identified the following three bottleneck challenges (and directions) for future works.

A. Towards Sophisticated Mobility Model

For tractability, our wireless modeling analysis (Sec. IV-A) assumes a simple two state transmission failures model to track channel errors by approximating the specific characteristics of autonomous vehicle. A more sophisticated time-varying channel model coupled with the mobility of vehicles using multiple states would require the tracking of the evolution of the channel state over time for each vehicle. To that end, a multidimensional Markov model needs to be developed, which requires further investigations.

Despite its simplicity, the two-state error model is expected to provide approximate results in vehicular networks for the following reason. In the first place, in the literature on mobile performance analysis that the use of Markovian channel modeling (intended to capture the details of the channel states and their mobility correlations) has been thoroughly studied. It turns out that such models hardly improve the estimates of the relevant performance metrics; see e.g. [42], [43]. Moreover, starting with a more detailed bit-level Markov channel

model, it is concluded in [42], [43] that the loss process as perceived by higher layer at a packet level is approximately an *independent and identically distributed* process.

Nevertheless, in our earlier works [35], [43], we already paid significant attention to the choice of the underlying channel modeling, as a result, we have experienced that using more complex models (i.e., models with correlated channel states and generally distributed service-times) hardly affects the estimated performance. Clearly, based on our previous observations, and in general, the packet-level loss process cannot be assumed completely i.i.d., particularly in situations when handovers, slow fading or shadowing plays an important role in autonomous vehicles. This requires an intelligent pragmatic approach, which is challenging and is left for future work.

B. Mobility Aware Efficient Verification

One of the important insights from this paper is that both FL and blockchain mechanism focus on consensus problems, whereby their pragmatic integration has the potential for a synergistic effect. However, one major challenge in practice is *forking*. Due to inevitable network delays, most often a vehicle may not have the most recent block, and may therefore create a different chain that branches from the main. Such forking reduces throughput, as only a single chain survives, eventually, while all other blocks in the different chains are dropped. Fortunately, for such energy-limited edge devices and time-sensitive applications, we have started investigating a lightweight idea along the line of [44] and tightly coupled with the celebrated mobility models of autonomous vehicles (discussed earlier in Sec. VII-A), which will considerably ameliorate the informational imbalance and speed up the mining verification process [44, Theorem 2]. This is because of the seminal *proof-of-work* mechanism, in which the consensus is reached through intensive mining processes has several limitations, such as, energy inefficiency, latency, and vulnerability to threats. In essence, our enhancement based on *proof-of-stake* mechanism will be a simple, lightweight approach improving latency significantly without changing the underlying consensus protocol and network parameters substantially. To this goal, several research questions remain poorly understood, in particular, with regards to i) analyzing adversarial behavior in such a polling approach and ii) a *stake reward for a global loyalty*. Both of these questions introduce their own theoretical and practical challenges to be addressed.

C. Privacy Leakage Risk Analysis

Looking forward, with the ever-evolving complexity of modern computational approaches violating user privacy, risk analysis of user privacy designed algorithms will provide a dynamically scalable approach to defend user privacy from the increasingly growing multidimensional interactions that big data-sets build. Using such risk aware privacy models at the heart of a FL environment will enable the whole network to learn from data in a robust decentralized way. Therefore, risk analysis of privacy leakage when miners try to verify their local models requires immediate attention. By harnessing the ever-increasing computational power on autonomous vehicles, and combining it with a secure and scalable, global learning session backed by a blockchain network with the ability to ensure oVML privacy, risk aware privacy models will enable any connected autonomous vehicle to participate and contribute data to a global privacy preserving, data sharing network even allowing the network to reward them in return.

VIII. Concluding Remarks

In this paper, we have enhanced federated learning with blockhain for the performance and privacy of autonomous vehicles. Our BFL framework facilitates efficient communication of autonomous vehicles, where local oVML learning modules exchange and verify their updates in a fully decentralized fashion. BFL has successfully enabled oVML machine learning without any centralized coordination by exploiting the consensus mechanism of blockchain. Furthermore, we developed a comprehensive analysis of BFL network system dynamics for the end-to-end system delay, which provides important insights for deriving optimal block arrival rate. The joint consideration of communication latency and consensus delay has been useful here. More importantly, the (offline and online) optimization approach based on our pragmatic analysis strikes a good balance between leaving out specific model details (to facilitate the numerical evaluation) and incorporating the crucial system features (to enable the systematic assessment of various non-trivial phenomena).

Finally, we conclude with a summary of the simulation and numerical findings. We have observed that there are optimal pairs $(\Lambda^\star, \tau_{\rm out})$ in the sense that the system delay $\mathbb{E}_{(n,m)}[T]$ is minimal with that value of $(\Lambda, \tau_{\rm out})$. Clearly, on one hand, one should use a sufficiently large number of retransmission to combat transmission failures, but on the other hand R should not be so large that the repeated transmissions and the resulting service time jointly increase $\mathbb{E}_{(n,m)}[T]$. Also important is the choice of an optimal block/update size, which will limit the increase in the overall delay but; a detailed study of this effect will be the topic of our future research.

References

[1] S. R. Pokhrel and J. Choi, "A decentralized federated learning approach for connected autonomous vehicles," in *Proc. IEEE WCNC Wkps*, to be published.

[2] D. Feng *et al.*, "Toward ultrareliable low-latency communications: Typical scenarios, possible solutions, and open issues," *IEEE Veh. Technol. Mag.*, vol. 14, no. 2, pp. 94–102, Jun. 2019.

[3] Y. Sun, L. Zhang, G. Feng, B. Yang, B. Cao, and M. A. Imran, "Blockchain-enabled wireless Internet of Things: Performance analysis and optimal communication node deployment," *IEEE Internet Things J.*, vol. 6, no. 3, pp. 5791–5802, Jun. 2019.

[4] J. Konečnỳ, H. B. McMahan, F. X. Yu, P. Richtárik, A. T. Suresh, and D. Bacon, "Federated learning: Strategies for improving communication efficiency," 2016, *arXiv:1610.05492*. [Online]. Available: http://arxiv.org/abs/1610.05492

[5] W. Y. B. Lim *et al.*, "Federated learning in mobile edge networks: A comprehensive survey," 2019, *arXiv:1909.11875*. [Online]. Available: http://arxiv.org/abs/1909.11875

[6] W. Chen *et al.*, "Cooperative and distributed computation offloading for blockchain-empowered industrial Internet of Things," *IEEE Internet Things J.*, vol. 6, no. 5, pp. 8433–8446, Oct. 2019.

[7] J. Konečný, H. B. McMahan, D. Ramage, and P. Richtárik, "Federated optimization: Distributed machine learning for on-device intelligence," 2016, *arXiv:1610.02527*. [Online]. Available: http://arxiv.org/abs/1610.02527

[8] S. Samarakoon, M. Bennis, W. Saad, and M. Debbah, "Distributed federated learning for ultra-reliable low-latency vehicular communications," 2018, *arXiv:1807.08127*. [Online]. Available: http://arxiv.org/abs/1807.08127

[9] J. Park *et al.*, "Distilling on-device intelligence at the network edge," 2019, *arXiv:1908.05895*. [Online]. Available: http://arxiv.org/abs/1908.05895

[10] M. Chen, Z. Yang, W. Saad, C. Yin, H. V. Poor, and S. Cui, "A joint learning and communications framework for federated learning over wireless networks," 2019, *arXiv:1909.07972*. [Online]. Available: http://arxiv.org/abs/1909.07972

[11] J. Kang, Z. Xiong, D. Niyato, H. Yu, Y.-C. Liang, and D. I. Kim, "Incentive design for efficient federated learning in mobile networks: A contract theory approach," in *Proc. IEEE VTS Asia Pacific Wireless Commun. Symp. (APWCS)*, Aug. 2019, pp. 1–5.

[12] M. Kuderer, S. Gulati, and W. Burgard, "Learning driving styles for autonomous vehicles from demonstration," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, May 2015, pp. 2641–2646.

[13] S. Niknam, H. S. Dhillon, and J. H. Reed, "Federated learning for wireless communications: Motivation, opportunities and challenges," 2019, *arXiv:1908.06847*. [Online]. Available: http://arxiv.org/abs/1908.06847

[14] S. Wang *et al.*, "Adaptive federated learning in resource constrained edge computing systems," *IEEE J. Sel. Areas Commun.*, vol. 37, no. 6, pp. 1205–1221, Jun. 2019.

[15] C. Decker and R. Wattenhofer, "Information propagation in the bitcoin network," in *Proc. IEEE P2P*, Sep. 2013, pp. 1–10.

[16] H. Kim, J. Park, M. Bennis, and S.-L. Kim, "Blockchained on-device federated learning," *IEEE Commun. Lett.*, early access, Jan. 10, 2019, doi: 10.1109/LCOMM.2019.2921755.

[17] I. Eyal and E. G. Sirer, "Majority is not enough: Bitcoin mining is vulnerable," *Commun. ACM*, vol. 61, no. 7, pp. 95–102, 2018.

[18] A. M. Antonopoulos, *Mastering Bitcoin: Unlocking Digital Cryptocurrencies*. Newton, MA, USA: O'Reilly Media, 2014.

[19] H. Seo, J. Park, M. Bennis, and W. Choi, "Communication and consensus co-design for low-latency and reliable industrial IoT systems," 2019, *arXiv:1907.08116*. [Online]. Available: http://arxiv.org/abs/1907.08116

[20] R. Bowden, H. P. Keeler, A. E. Krzesinski, and P. G. Taylor, "Block arrivals in the bitcoin blockchain," 2018, *arXiv:1801.07447*. [Online]. Available: http://arxiv.org/abs/1801.07447

[21] M. Shen, X. Tang, L. Zhu, X. Du, and M. Guizani, "Privacy-preserving support vector machine training over blockchain-based encrypted IoT data in smart cities," *IEEE Internet Things J.*, vol. 6, no. 5, pp. 7702–7712, Oct. 2019.

[22] G. Kumar, R. Saha, M. K. Rai, R. Thomas, and T.-H. Kim, "Proof-of-Work consensus approach in blockchain technology for cloud and fog computing using maximization-factorization statistics," *IEEE Internet Things J.*, vol. 6, no. 4, pp. 6835–6842, Aug. 2019.

[23] S. Wang, C. Wang, and Q. Hu, "Corking by forking: Vulnerability analysis of blockchain," in *Proc. IEEE Conf. Comput. Commun. (INFOCOM)*, Apr. 2019, pp. 829–837.

[24] M. Chase, "Multi-authority attribute based encryption," in *Proc. Theory cryptogr. Conf.* Berlin, Germany: Springer, 2007, pp. 515–534.

[25] H. Shrobe, D. L. Shrier, and A. Pentland, "Enigma: Decentralized computation platform with guaranteed privacy," in *New Solutions for Cybersecurity*. Mainz, Germany: MITP, 2018, ch. 15, pp. 425–454.

[26] C. Natoli, J. Yu, V. Gramoli, and P. Esteves-Verissimo, "Deconstructing blockchains: A comprehensive survey on consensus, membership and structure," 2019, *arXiv:1908.08316*. [Online]. Available: http://arxiv.org/abs/1908.08316

[27] S.-H. Kim and W. Whitt, "Choosing arrival process models for service systems: Tests of a nonhomogeneous Poisson process," *Nav. Res. Logistics*, vol. 61, no. 1, pp. 66–90, Feb. 2014.

[28] L. Brown *et al.*, "Statistical analysis of a telephone call center: A queueing-science perspective," *J. Amer. Stat. Assoc.*, vol. 100, no. 469, pp. 36–50, Mar. 2005.

[29] F. Sattler, S. Wiedemann, K.-R. Müller, and W. Samek, "Robust and communication-efficient federated learning from non-IID data," 2019, *arXiv:1903.02891*. [Online]. Available: http://arxiv.org/abs/1903.02891

[30] H. B. McMahan, D. Ramage, K. Talwar, and L. Zhang, "Learning differentially private recurrent language models," 2017, *arXiv:1710.06963*. [Online]. Available: http://arxiv.org/abs/1710.06963

[31] J. Göbel, H. P. Keeler, A. E. Krzesinski, and P. G. Taylor, "Bitcoin blockchain dynamics: The selfish-mine strategy in the presence of propagation delay," *Perform. Eval.*, vol. 104, pp. 23–41, Oct. 2016.

[32] M. Rosenfeld, "Analysis of hashrate-based double spending," 2014, *arXiv:1402.2009*. [Online]. Available: http://arxiv.org/abs/1402.2009

[33] M. Rosenfeld, "Analysis of bitcoin pooled mining reward systems," 2011, *arXiv:1112.4980*. [Online]. Available: http://arxiv.org/abs/1112.4980

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

[35] S. R. Pokhrel and M. Mandjes, "Improving multipath TCP performance over WiFi and cellular networks: An analytical approach," *IEEE Trans. Mobile Comput.*, vol. 18, no. 11, pp. 2562–2576, Nov. 2019.

[36] X. Li, K. Huang, W. Yang, S. Wang, and Z. Zhang, "On the convergence of FedAvg on non-IID data," 2019, *arXiv:1907.02189*. [Online]. Available: http://arxiv.org/abs/1907.02189

[37] T. Li, A. K. Sahu, M. Zaheer, M. Sanjabi, A. Talwalkar, and V. Smith, "Federated optimization in heterogeneous networks," 2018, *arXiv:1812.06127*. [Online]. Available: http://arxiv.org/abs/1812.06127

[38] Z. Yan, P. Zhang, and A. V. Vasilakos, "A survey on trust management for Internet of Things," *J. Netw. Comput. Appl.*, vol. 42, pp. 120–134, Jun. 2014.

[39] M. R. Palattella *et al.*, "Internet of Things in the 5G era: Enablers, architecture, and business models," *IEEE J. Sel. Areas Commun.*, vol. 34, no. 3, pp. 510–527, Mar. 2016.

[40] G. Lee, J. Park, W. Saad, and M. Bennis, "Performance analysis of blockchain systems with wireless mobile miners," 2019, *arXiv:1906.06759*. [Online]. Available: http://arxiv.org/abs/1906.06759

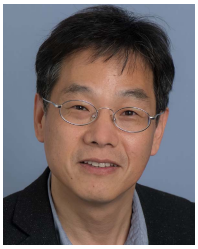
[41] S. Kim, "Impacts of mobility on performance of blockchain in VANET," *IEEE Access*, vol. 7, pp. 68646–68655, 2019.

[42] D. Moltchanov and R. Dunaytsev, "Modeling TCP performance over wireless channels with a semi-reliable data link layer," in *Proc. 11th IEEE Singap. Int. Conf. Commun. Syst.*, Nov. 2008, pp. 912–918.

[43] M. Panda, H. L. Vu, M. Mandjes, and S. R. Pokhrel, "Performance analysis of TCP NewReno over a cellular last-mile: Buffer and channel losses," *IEEE Trans. Mobile Comput.*, vol. 14, no. 8, pp. 1629–1643, Aug. 2015.

[44] G. Fanti, J. Jiao, A. Makkuva, S. Oh, R. Rana, and P. Viswanath, "Barracuda: The power of ℓ-polling in proof-of-stake blockchains," in *Proc. 20th ACM Int. Symp. Mobile Ad Hoc Netw. Comput. (Mobihoc)*, 2019, pp. 351–360.

Shiva Raj Pokhrel (Member, IEEE) received the B.E. and M.E. degrees from Pokhara University, Nepal, in 2007 and 2013, respectively, and the Ph.D. degree from the Swinburne University of Technology, Australia, in 2017. He was a Research Fellow with the University of Melbourne from 2017 to 2018 and a Network Engineer with Nepal Telecom from 2007 to 2014. He is currently a Lecturer of mobile computing with Deakin University, Geelong, Australia. His research interests include federated learning, blockchain modeling, optimization, recommender systems, beyond 5G, cloud computing, dynamics control, Internet of Things, and cyber-physical systems and their applications in smart manufacturing, autonomous vehicles and cities. He was a recipient of the prestigious Marie Skłodowska-Curie Grant Fellowship in 2017.



Jinho Choi (Senior Member, IEEE) was born in Seoul, South Korea. He received the B.E. degree *(magna cum laude)* in electronics engineering from Sogang University, Seoul, in 1989, and the M.S.E. and Ph.D. degrees in electrical engineering from the Korea Advanced Institute of Science and Technology (KAIST) in 1991 and 1994, respectively. In 2018, he was a Professor/Chair in wireless with Swansea University, U.K., and also a Professor with the Gwangju Institute of Science and Technology (GIST), South Korea. He is currently with the School of Information Technology, Burwood, Deakin University, Australia, as a Professor. He authored two books published by Cambridge University Press in 2006 and 2010. His research interests include the Internet of Things (IoT), wireless communications, and statistical signal processing. He received the 1999 Best Paper Award for Signal Processing from EURASIP and the 2009 Best Paper Award from WPMC (Conference). He is currently an Editor of IEEE TRANSACTIONS ON COMMUNICATIONS and IEEE WIRELESS COMMUNICATIONS LETTERS and a Division Editor of *Journal of Communications and Networks* (JCN). He had served also as an Associate Editor or Editor of other journals including IEEE COMMUNICATIONS LETTERS, JCN, IEEE TRANSACTIONS ON VEHICULAR TECHNOLOGY, and ETRI journal.