Electric Vehicle Charging Queue Management with Blockchain

Subhasis Thakur(✉) and John G. Breslin

Data Science Institute, National University of Ireland Galway,
University Road, Galway H91 TK33, Ireland
{subhasis.thakur,john.breslin}@nuigalway.ie

Abstract. We will have to build adequate charging station infrastructure to support the massive proliferation of electric vehicles over the next few years. A federated structure for charging stations can serve as a solution for building a large number of charging stations. It will allow microgrids, private houses, and hotels to provide charging services. Present literature on the topic of managing electric vehicle charging queues has not addressed the problems associated with such a federated network of charging stations. In this paper, we solve the electric vehicle charging queue management problem through a federation of charging stations using blockchains. In this electric vehicle queue management solution, we show that (a) stations cannot hide information to manipulate charging queues, (b) it enhances the privacy of electric vehicles as they are not required to reveal their exact desired recharging locations, (c) it encourages electric vehicles to recharge at the prescribed charging stations, (d) it allocates better recharging stations to electric vehicles who reveal their exact desired recharging location, i.e., one pays for privacy, (e) it supports load balancing over stations, and (f) blockchain not only provides a transparent and secure solution but also provides incentives to the station owners to bear the cost of establishing a charging station.

Keywords: Electric vehicle · Charging station · Blockchain

1 Introduction

The electrification of transportation plays an important role in reducing greenhouse gas emissions as transportation accounts for 23% of greenhouse gases. Governments are starting to ban cars with internal combustion engines by as early as 2030. These policies will introduce 150 million units of Electric Vehicles (EVs) and 400 million electric two-wheelers by 2030, with 1.2 billion EVs by 2060. The availability of charging stations is a critical component for rapid EV adoption. We will need a large number of charging stations to ensure availability to a massive number of EVs.

A federation of charging stations is a potential solution to support the required growth of charging stations. It is not only large firms who can participate in establishing charging stations but small microgrids, private residential

buildings, shops, hotels, bed and breakfasts, rural firms, and others should also be encouraged to establish charging stations. Often these stations will be owned by private owners in remote locations supported by renewable energy resources. In this paper, we focus on such a federated structure of charging stations. The problems with federating charging stations are:

Trust: Stations may not trust each other to share information on their EV queues as each station will want to maximize its queue length.

Monitoring and Enforcement: We need monitoring and governance capabilities to monitor the price for recharging services and also to enforce governance rules to regulate EV queues for energy grid load balancing.

Computational Cost: Significant computational resources are needed to coordinate EV queue management for large road networks spanning over neighbouring countries such as in the European Union. Also, we need to assess the economic feasibility of developing the computational infrastructure required for the associated information processing and EV queue management.

The present literature on EV queue management does not address these problems. We have therefore made the following contributions in this paper: **(1)** We develop a platform for charging stations to interact among themselves and solve the EV charging scheduling problem in a collaborative fashion. Using blockchain, we develop a secure, traceable, distributed solution for the EV charging scheduling problem. **(2)** We develop a distributed charging scheduling algorithm which aims to minimize the waiting time for EVs to recharge and that also prevents EVs from manipulating the system. A federated charging station network is likely to be less regulated than a centralized station network. Hence in this federated environment a station will attempt to maximize its queue of EVs to increase its profit by not disclosing queue information to nearby stations. Our solution prevents stations from executing such selfish actions. **(3)** We develop a flexible charging schedule for EVs. The present solutions for EV charging schedules are restrictive in the sense that they allow limited stations for each EV to charge at. The feasibility of an EV to charge at a prescribed station depends on traffic conditions and the personal needs of the EV driver. Also an EV driver may not wish to reveal their exact destination due to privacy concerns. In this paper we assign a large number of stations in a local area to an EV. It improves the flexibility of the charging schedule and it then becomes more likely that an EV will comply with the prescribed schedule. **(4)** The proposed solution encourages EVs to remain compliant with the prescribed station schedule. **(5)** It allocates better recharging locations to EVs who have provided accurate desired recharging locations compared with other EVs who may not want to do so due to privacy concerns. **(6)**The proposed solution supports load balancing over stations. **(7)** It continuously changes the flexibility of allocation, i.e., if the stations are not congested then EVs will get more options to recharge.

The paper is organized as follows. In Sect. 2 we discuss related literature. In Sect. 3 we present a formal model for federated charging stations. In Sect. 4 we present a blockchain-based solution for the EV charging scheduling problem.

In Sect. 5 we present an experimental evaluation of the proposed solution. We conclude the paper in Sect. 6.

2 Related Literature

[7] solves the problem of charging price manipulation by cartels of charging stations by maximizing the number of owners of charging stations in each locality. [9] develops a pricing model for charging with load balancing based on frequency regulation signals from the electricity grid. [11] solves the EV queue management problem using coalitional game theory. [5] solves the EV queue management problem with load balancing among stations while minimizing the waiting time of EVs. [2] uses a genetic algorithm to solve the EV charging scheduling problem. [10] develops a pricing model considering travel patterns, EV driver behaviors, and traffic information. [6] presents a survey on economy-driven approaches such as auctions, Stackelberg games, and other potential games to solve the EV charging scheduling problem. Existing solutions for the EV scheduling problem do not address issues around the federation of stations, and they ignore the problem of developing computational infrastructure for information processing in such a federation of stations. The shortcomings of existing solutions for EV charging queue management in a set of federated charging stations are as follows: **(1)** Existing solutions do not address the requirement to evaluate trust among charging stations and EVs. **(2)** Existing solutions do not provide a secure platform for interaction. Most of these solutions assume that such a platform exists. **(3)** Existing solutions do not provide any monitoring or enforcement capabilities. **(4)** Existing solutions do not provide any incentive to EVs to remain compliant with the prescribed solution. **(5)** Existing solutions do not provide any incentive to the station owners to bear the cost of establishing charging station infrastructure and computation infrastructure. **(6)** Existing solutions are not very flexible as they only offer a small number of stations per EV for recharging. In this paper we propose a blockchain-based EV scheduling solution. We use a proof-of-work-based blockchain as developed in [4].

3 Federated Charging Station Network

Definition 1 *(Road network): $G = (V, E)$ be an undirected graph representing a road network with nodes V (landmarks in a city) and edges E (road segments connecting two landmarks). $D(E_i \in E)$ (positive integer) will denote the length of the road segment E_i.*

Next we define a charging station network as an undirected graph where each node represents a charging station and stations are within an average distance that a fully charged EV can travel. The intuition behind such a graph formation is that an EV can design its travel path according to such a graph as it may need multiple recharging stops.

Definition 2 *(Charging station network): $H = (N, L)$ be an undirected graph representing a charging station network where, (1) N is a set of k charging stations and L is the set of links among the charging stations. (2) $Loc(N_i) \in V$ will denote the location of the station $N_i \in N$. The location of each station is unique and one landmark $V_i \in V$ will accommodate only one charging station. (3) A link L_i among two stations N_i and N_j is a path P in the road network G such that P connects $Loc(N_i)$ and $Loc(N_j)$ and the length of the path is less than or equal to Δ (Positive integer) i.e., $\sum_{E_i \in P} D(E_i) \leq \Delta$.*

Definition 3 *(EV): We define the properties of an EV $C_i \in C$ (C is the set of all EVs) as follows: (1) $Start(C_i) \in t$ be the starting time of the EV. (2) $Source(C_i) \in V$ be the starting location of C_i and $Sink(C_i) \in V$ be the destination of C_i. (3) $Battery(C_i, t_j) \in [0, 100]$ will denote the level of battery charge of the EV C_i at time t_j. (4) $\Theta(x) \in [0, 100]$ (x is positive integer) will denote the decrement of charge if the EV has travelled a distance x since its last charge.*

Definition 4 *(Traffic): We define the traffic as follows: (1) C is a set of m EVs. (2) $t = (t_1, t_2, \dots)$ be a discrete successive time sequence. For example t_1 is 6 AM, t_2 is 7 AM and so on. (3) The traffic load at time t_i is given by the function $\theta(t_i)$ (positive integer), i.e., $\theta(t_i) \subseteq C - \cup_{x=0}^{x=i-1} \theta(t_x)$.*

Definition 5 *(Queue): $Q(N_i, t)$ will denote the queue of EVs who want to charge at location N_i in a time t. $Q(N_i, t) \in 2^C$, i.e., it is a sequence of EVs.*

We assume the following: (1) An EV chooses the shortest path (in the graph G) to travel from its source to destination. (2) An EV chooses to charge once the level of charge in its battery reaches $\rho \in [0, 100]$ and it cannot complete its journey with the remaining charge. In the experimental evaluation we assume that ρ is 40. We consider multiple owners of the stations. An owner may only own a subset of stations.

Definition 6. *There are z firms $f_1, \dots, f_z$ who provide charging services, i.e., who own charging stations. $F(f_i) \subset N$ will denote the charging stations owned by a firm f_i.*

It is assumed that the station locations for each firm are chosen uniformly at random. Now we define a partition over charging stations as groups of connected subgraphs. Each such group of stations are options given to an EV to recharge. An EV may charge at any station in such a group.

Definition 7. *A station segmentation is a partition over the charging stations into K (positive integer) groups denoted as $\Pi = (\Pi_i, \dots, \Pi_k)$ such that: (1) a station only belongs to one group $\forall \Pi_i, \Pi_j, \Pi_i \cap \Pi_j = \emptyset$ and $\bigcup_{i \in [1,k]} \Pi_i = N$, (2) the induced subgraph for each group of stations Π_i on the charging station network is a connected graph.*

Definition 8. *A station allocation function δ maps an EV to a station segmentation group, i.e., $\delta : C \mapsto \Pi$.*

Definition 9. *A station segmentation Π is stable if there is a station allocation function δ such that the following conditions hold: (1) Any two stations in a group are at most d distance apart, i.e., $\forall \Pi_i$, $\forall N_x, N_y \in \Pi_i D(N_x, N_y) \leq d$. (2) The load difference between any two groups is at most ϵ_1 (positive number), i.e.,*

$$\forall \Pi_i, \Pi_j \in \Pi, ABS(|\cup_{c_i:\delta(c_x)=\Pi_i} c_x| - |\cup_{c_i:\delta(c_x)=\Pi_j} c_x|) \leq \epsilon_1$$

where ABS gives an absolute number and $||$ is a count function. (3) The waiting time difference between any two groups is at most ϵ_2 (positive number) i.e.,

$$\forall \Pi_i, \Pi_j \in \Pi, ABS(|\sum_{N_x \in \Pi_i} Q(N_x, t_a)| - |\sum_{N_x \in \Pi_j} Q(N_x, t_a)|) \leq \epsilon_2.$$

The explanations for the above conditions are as follows: *[Condition 1:]* This ensures that stations in each group are within a fixed proximity. Note that an EV will be asked to charge at any station in such a group of stations. Hence stations in each group should be in close proximity. *[Condition 2:]* This ensures that the difference between the number of EVs who will charge in any two groups of stations is bounded by a fixed number. Note that in a federated station environment, each firm wants to maximize the number of EVs who use their charging stations. Hence this condition ensures that there is no firm that gets more EVs. *[Condition 3:]* It ensures that the difference between the waiting time for two EVs who will recharge at different groups of stations is bounded. Hence it makes sure that the benefit of EVs (in terms of how long they will wait to get recharged) is bounded. Thus there should not be any preference over groups of stations as EVs will all require an almost equal amount of time to recharge their battery.

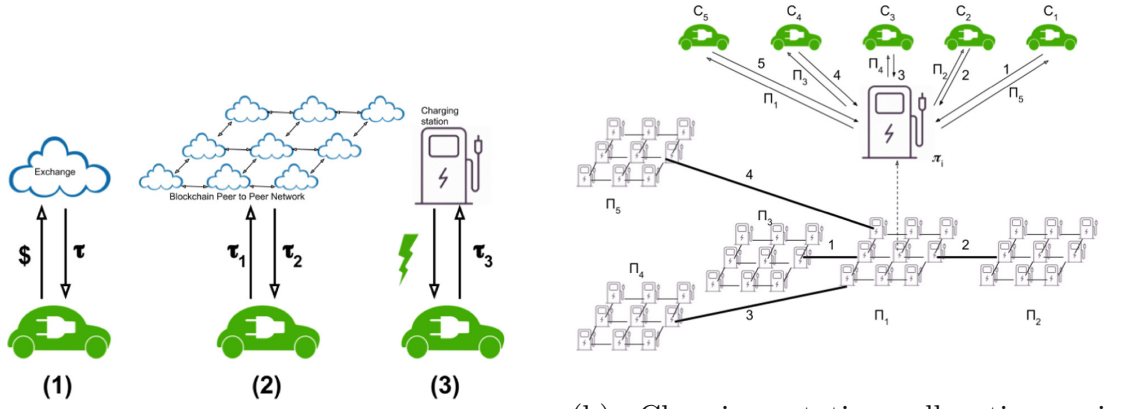
Definition 10. *A sequence of stable station segmentations in a time duration $[t_1, t_2, \ldots, t_x]$ is a sequence of station segmentations $\Pi^1, \Pi^2, \ldots, \Pi^x$ such that Π^1 is stable at time t_1, Π^2 is stable at time t_2 and so on.*

We solve the sequence of stable station segmentations problem with blockchain.

4 EV Scheduling with Blockchain

The blockchain (BC) mechanism works as follows: **(1)** BC allows peers of a peer-to-peer network to transfer tokens among themselves using transactions. **(2)** If a peer P_1 wants to send x tokens to P_2 then it creates the transaction T_1 and announces it to its neighbours in the BC peer-to-peer network. **(3)** Once such a neighbour P_3 receives the transaction T_1, P_3 attempts to verify it. If it can verify T_1 as a valid transaction then it forwards T_1 to its neighbours. **(4)** BC stores consistent replicas of the transaction history on multiple peers. Valid transactions are grouped into a block and blocks are stored in a chain of blocks where each block has only one parent block. **(5)** A new block can be added to the

BC as the child of the most recent block. Any peer can verify transactions and add a new block to the BC provided it satisfies the conditions of the distributed consensus protocol. **(6)** The distributed consensus protocol ensures all peers have the same replica of the BC.



(a) EV scheduling with Blockchain

(b) Charging station allocation using blockchain transactions.

Fig. 1. (a) EV scheduling with Blockchain, (b) Charging station allocation using blockchain transactions.

As illustrated in Fig. 1, EV scheduling with blockchain works as follows: **(1)** An EV buys a set of tokens from an exchange using other types of currency. Tokens are used to find charging stations and to pay stations for recharging. **(2)** An EV sends x amount of tokens (in transaction τ_1) to a nearby charging station to find a suitable charging station. We initialize the queue management procedure by a partition over the stations and a score for each station segment that indicates the level of load on it. **(3)** Once a station receives a request from an EV to find a charging station, it evaluates if the score of its own station segment is the least when compared with surrounding station segments. It creates a transaction τ_2 whose input is τ_1 and labels it with stations from one of the surrounding station segments (including itself) which has the least score. Such a label indicates where the EV should recharge. **(4)** Once the EV receives τ_2, it chooses a station (C_x) from the list of stations mentioned in τ_2 and recharges itself by sending a new transaction τ_3 (whose input is τ_2) to it. Now we present detailed description this solution.

4.1 Blockchain Infrastructure

We assume that EVs have access to cloud-based blockchain wallets. Advances in vehicular networks, the vehicular cloud, and roadside communication facilities [1,8] justify this assumption. Every station owner is required to commit certain computational resources, i.e., they work as miners. Each station has a node in the blockchain peer-to-peer network. The identity of stations, i.e., nodes in the blockchain, is recognized with its public key, and the GPS coordinates for each station are stored in the identification information for each node.

4.2 Transactions

Blockchain uses an unspent transaction output (UTXO) data structure to express transactions. According to this, inputs to a transaction must be unspent, i.e., not used as the input to any other transaction. It ensures that funds cannot be double spent. We refer to the Bitcoin Wiki's Transaction page[1] for a detailed description of the blockchain transaction data structure. We have a few extra fields in our transaction data structure, which are (1) "Coordinates": GPS location of the creator of the transaction, (2) "Evaluated": Takes a value of True or False, and (3) "Stations": Either empty or public keys of a set of charging stations. The procedure of charging station allocation is as follows:

(1) First an EV, say C_1 gathers x amount of tokens τ. It can do so by converting any other currency into τ from a token exchange similar to Bitcoin exchanges. One token τ will represent the cost for 1 kWh. C_1 may evaluate the approximate number of tokens it needs to recharge.
(2) Once C_2 wants to find a recharging station, it first finds a nearby charging station using its own GPS location and the GPS locations of charging stations. A web service on the blockchain can be used to search for all nearby stations. Let such a station be π_i. C_1 constructs a transaction τ_x of amount x and sends it to π_i with "Coordinates" as its own GPS location, "Evaluated" as False and "Stations" as empty.
(3) The station π_i may receive many such transactions within a short interval. The interval time is fixed. It may receive transactions from C_1, C_2, C_3, C_4, C_5 in one interval. π_i will perform the station allocation operation.
(4) π_i is part of the station segment Π_1 which has neighbouring station segments, e.g., $\Pi_2, \Pi_3, \Pi_4, \Pi_5$ at a distance of 2, 1, 3, 4. The distance between two station segments is measured as the length of the shortest path that connects two stations (one in each segment) who are the most centrally located in each respective station segment. Also, each station segment is assigned a score with the following equation

$$Score(\Pi^i) = Sin(\frac{\sum_{N_x \in \Pi_i}(3.14 \times |Q_x|)/(2 \times Limit_2)}{\text{Number of stations in } \Pi_i}).$$

$Limit_2$ is the maximum allowed queue size at any station. Q_x is the queue at station N_x. The $Score$ for a station segment generates a number between 0 and 1 which reflects the average level of congestion at the stations in the station segment. Using Algorithm 1, C_i allocates stations to C_1, C_2, C_3, C_4, C_5 in such a way that it balances congestion across different station segments and it also allocates stations close to π_i to the EVs whose transaction amount is high.
(5) Algorithm 1 is as follows: Let π_i execute Algorithm 1. First it computes the amount of free spaces available in surrounding station segments. Next it allocates stations to EVs in the order of decreasing amounts of transaction

[1] https://en.bitcoin.it/wiki/Transaction.

value. It chooses a station for an EV in the order of increasing distance from its own station segment. Note that Algorithm 1 can be executed by several blockchain nodes in a parallel and asynchronous fashion as the station requests from EVs located long distances away need not be included as input to Algorithm 1.

(6) After π_i computes a station allocation using Algorithm 1: for each allocation, say "Allocate Π_x to C_i", it creates a transaction τ_i' whose recipient is C_i, "Evaluated" is True, "Stations" is the public keys of all stations in the segment Π_x, the input transaction is τ_i (which it has received from C_i), and the amount is $Amount(\tau_i) - TransactionFee$ where $TransactionFee$ is a small predefined value which π_i keeps to itself as a reward for processing the station allocation procedure.

(7) After C_i receives τ_i', it chooses any station, say π_x, mentioned in the "Stations" of τ_i'. C_i recharges and pays π_x with a transaction τ_i'' whose recipient is π_x, input is τ_i', "Evaluated" is False, "Stations" is empty and transaction amount is calculated according to the energy transferred from π_x to C_i.

(8) It is compulsory that C_i uses any station from the station list mentioned in τ_i' to use the token τ_i'. Hence EVs will remain compliant with having to use the prescribed charging stations.

(9) Algorithm 1 allocates the closest station segment to the EV who made the biggest transaction. Note that an EV can only use the token at prescribed stations and hence, the risk of using a high value transaction expresses an EV's desire to recharge at stations closest to π_i. By doing so the EV loses privacy but gets rewarded with a close recharging station. Hence the level of privacy is expressed using the value of the transaction.

Algorithm 1. Station allocation

Data: Station segmentation $\Pi = \Pi_1, \ldots \Pi_k$, x Transactions $\tau_1, \ldots, \tau_x$ from $C' = C_1, \ldots, C_x$ to $\pi_i \in \Pi_1$ who executes the algorithm

Result: Station allocation

begin

$\Pi' = (\Pi_1, \ldots, \Pi_a)$ be neighbouring station segments of Π_1, $D \leftarrow$ distance of Π' from Π_1, $FreeSpace \leftarrow$ vector of length a denoting available charging spaces in each $\Pi_x \in \Pi'$ by using $Score$ function, $FreeSpace' \leftarrow$ vector of length a denoting fraction of x EVs that each $\Pi_x \in \Pi'$ will accommodate

for *Each* $C_i \in C'$ *in decreasing order of transaction amount* **do**

for *Each* $\Pi_x \in \Pi'$ *in increasing order of distance* D **do**

if C_i *not allocated and* Π_x *has space* **then**

Allocate Π_x to C_i, $Freespace'[x] = Freespace'[x] - 1$

4.3 Distributed Consensus Protocol

The distributed consensus protocol ensures that peers of a BC peer-to-peer network reach consensus about the validity of a transaction, i.e., they all agree that the transaction is valid or invalid. Also, miners compete to add new blocks to the BC as there is a financial reward for doing so. The distributed consensus protocol determines the winner of such a race to add new blocks to the BC. It may happen that two or more miners may add a new block almost at the same time. Such an event creates a fork in the BC. The distributed consensus protocol eliminates such a fork. There are two major types of distributed consensus protocols: proof of work [4] and proof of stake [3]. In proof of work, a miner has to solve a puzzle before it can add a new block to the BC. In proof of stake, the protocol uses low complexity puzzles and peers are regularly rewarded based on their stake. Briefly the protocol is as follows:

(1) Each miner maintains a BC head which is the block whose distance (shortest path) from the first block is the maximum distance. **(2)** If a miner gets a new block (say B) it uses this procedure: (a) It checks if the new block is a valid block. If B is valid then the miner follows either of the next two steps and forwards the block to its neighbours. (b) If the parent block of the new block is the most recent block of the BC then it adds B as its child and recognizes B as the BC head. (c) If the parent block (say A) of the new block is not the most recent block in the BC then it adds B as a child of A. But it does not change the BC head. **(3)** If it creates a new block (say B) then it adds B as a child of the current BC head and recognizes B as the BC head. **(4)** If at any time, the block whose distance from the first block is the maximum distance, it is recognized as the BC head. Beside block creation and verification according to the proof of stake protocol, our miners are required to solve station segmentation adjustment problems before publishing a new block. It executes two procedures (a) "adjust station segmentation" and (b) "split or merge segments" as described in Algorithms 2 and 3.

In "adjust station segmentation", a miner swaps neighbouring stations between two station segments to reduce differences in scores among the station segments. The "split or merge segments" (the split method is shown in Algorithm 3) is performed according to historical changes in demands for charging stations. According to this, if demand is increasing at the time of creation of the block then the miner must perform a 'split' procedure as shown in Algorithm 3, otherwise it should merge station segments. The split procedure splits a station segment into two station segments in such a way that station segments induce a connected subgraph on the station graph (hence it is appropriate to allocate such segments as options for recharging an EV), and it separates/distributes stations with long queues among them (hence the resultant segments are evenly congested). We need to split a station segment to restrict EVs from choosing convenient stations (e.g. those close to the city centre) when the demand is high. It may happen that most of the EVs will choose convenient stations simultaneously, and it will not only congest certain areas but also have a negative impact on the electricity grid. The 'merge' procedure merges two neighbouring

segments into one if the size of the resultant segment is within a certain range and the maximum distance between any pair of stations in the segment is less than a particular limit. A miner must execute the merge procedure if the historical demand for stations is decreasing at the time of block creation. Thus when the demand is low EVs get more options to recharge. Note that the split and merge procedures are performed at predefined times, determined by historical traffic data. This means that if the historical traffic volume increases sharply from 7 AM to 11 AM then all blocks created during this time must execute a split procedure. Similarly if the historical traffic volume decreases sharply from 11 AM to 2 PM then all blocks created during this time must execute a merge procedure.

Finally the block creation rate will be controlled using the level of complexity of the puzzle that a miner has to solve before it publishes a new block. With higher changes in the demand for stations, we create blocks more frequently to adjust station segments. Note that such modifications of the complexity of a puzzle depends on historical traffic volume information. Also, it will be a predefined in this blockchain as to whether the complexity of a puzzle will be low or high. For example, if the historical traffic volume increases sharply from 7 AM to 11 AM then the complexity of a blockchain puzzle will be low to generate blocks more frequently and to perform station segmentation adjustments and split/merge procedures more frequently. Such changes in the blockchain complexity will be predefined based on historical traffic information.

Algorithm 2. Station segmentation adjustment

Data: A station graph $H = (N, L)$, Group size limits K_1, K_2, Distance limit D
Result: Adjusted station segmentation
begin
  Station segmentation $\Pi^1, \ldots, \Pi^z$, $Score \leftarrow$ assign score to segments, Π^+ segments with score more than $AVG(Score)$, Π^- segments with score less than $AVG(Score)$
  for *Each* $\Pi^i \in \Pi^-$ *in increasing order of score* **do**
    for *Each* $\Pi^j \in \Pi^+$ *in increasing order of score* **do**
      for *Each* $N_x \in \Pi^i$ *in increasing order of queue* **do**
        $N_y \in \Pi^j$ such that, N_x has an edge with $\Pi^j - N_y$ and N_y has an edge with $\Pi^i - N_x$
        Swap N_x and N_y
        if $Score(\Pi_i) \geq AVG(Score)$ **then**
          Break;

Algorithm 3. Split a station segment

Data: A station segment $\Pi_i = (\pi_1, \pi_k)$ with queues $(Q_1, \ldots, Q_k)$
Result: Split Π_i into two segments
begin
  $Order \leftarrow$ order stations by decreasing queue length, $Group_1 \leftarrow$ station with highest queue length, $Group_2 \leftarrow$ station with second highest queue length
  while *Stations not added to* $Group_1$ *or* $Group_2$ **do**
    $Neighbour_1 \leftarrow$ neighbouring stations of $Group_1$, $Neighbour_2 \leftarrow$ neighbouring stations of $Group_2$, $\pi^* \in Neighbour_1 \cap Neighbour_2$
    if *Distance from* π^* *to* $Group_1$ *is more than the same for* $Group_2$ **then**
      Add π_i^* to $Group_1$
    else
      Add π_i^* to $Group_2$
    Add $\pi_x \in Neighbour_1 - \pi^*$ to $Group_1$ if $Score(\pi_x)$ is minimum in $Neighbour_1$, Add $\pi_x \in Neighbour_2 - \pi^*$ to $Group_2$ if $Score(\pi_x)$ is minimum in $Neighbour_2$

4.4 Observations

We note the following with blockchain-maintained EV queue management. (a) Stations cannot hide information on EV queue as all transactions are visible to all peers of the blockchain, (b) Using experimental evaluation we will show waiting time for EVs is less with blockchain maintained EV queue management, (c) an EV can choose any station in a station segment and hence it enhances its privacy, (4) as EVs can only use token to pay the prescribed stations, it will remain compliant, (5) stations segments are revised for load balancing, (6) station allocation to EVs are solved in a distributed fashion and (7) the financial incentives as transaction fees and mint coins will draw investments to build the computational infrastructure for EV queue management information processing.

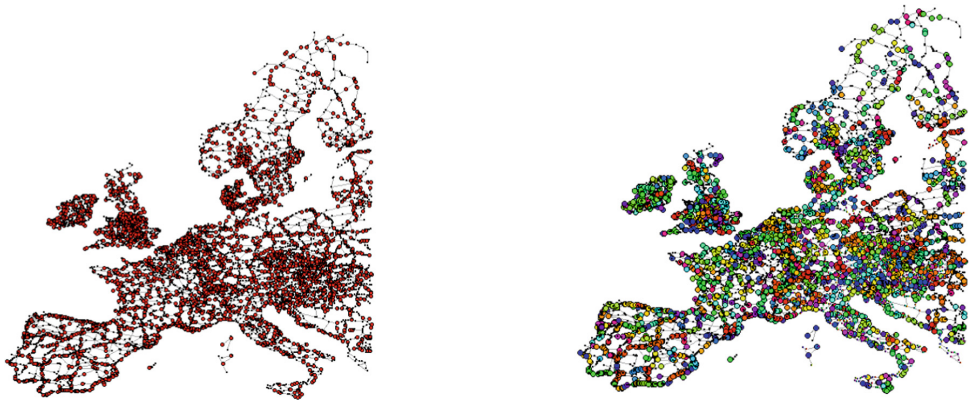
5 Experimental Evaluation

We use agent-based modelling to simulate the traffic network. Stations and EVs are modelled as agents. We use asynchronous event simulation in Python to implement the activities of station and EVs. The blockchain is also modelled within the same simulation as we include block creation and associated activities within the behaviors of EVs and stations. The simulation is as follows:

(1) The simulation starts with a random partition station segmentation Π^0. We only consider the distance among stations and the maximum number of stations allowed in each station group as the factors to determine Π^0. The intuition behind generating such a partition is that we can start usage of the blockchain-based solution at 12 AM when traffic loads are negligible and such segmentation is modified over time as the traffic load changes.
(2) At every iteration of the simulation, we introduce new cars according to historical traffic information and we execute the behaviour of EVs and stations in an asynchronous fashion.
(3) We consider the following behavior of EVs: An EV proceeds towards its destination until its charge falls below a certain limit, and after that it searches for a station and moves towards the station to get recharged. After that it continues its journey towards its destination. An EV always follows the shortest paths.
(4) We consider the following behavior of stations: Each station executes three tasks as independent processes: (a) it recharges EVs, (b) it assigns stations to an EV for recharging, and (c) it creates blocks by verifying transactions and modifying station segmentations, and it either performs a 'split' or 'merge' operation according to historical demand information.
(5) We simulate the rewarding process for miners according to the proof of stake procedure. The number of blocks created in each iteration depends on the historical demand information. We create more blocks as demand increases and vice versa. The probability that a miner will receive a minted token depends on its stake, i.e., the amount of tokens it owns and when it received its last minted token.

We use the road network for Europe from Open Street Map (https://www.openstreetmap.org/) to extract major roads and highways. We process 'Shape' files for such data in R using the libraries "igraph, sp, shp2graph, rgdal". The resultant road network contains 10527 nodes and 10100 edges with a mean edge distance of 17 km. We choose 3000 nodes as charging stations uniformly at random. The station network has 26294 edges where two stations are considered as neighbours if the distance between them is at most (0.1 * mean distance amongst all pairs of stations). The mean edge length of the station graph is 77 km, i.e., on average two stations are 77 Km apart from each other. Also, the mean degree of stations is 17. The initial partition over the stations has station segments with an average segment size of 3. In any such segment, the average maximum distance between two stations is 70 km. We use four datasets (we will refer to these as datasets 1, 2, 3, 4) with 2000 EVs, and we set their source and destination locations in four local areas (i.e., four connected subgraphs of the European road network) with a diameter of 500 km, i.e., the maximum distance between any

two nodes, and on average there are 150 charging stations. We set the speed of each EV as 50 km/h. Each iteration of the simulation corresponds to one hour. We set the range of EVs as 200 km. The source to destination distance for each EV is approximately 300 km. An EV searches for a charging station if (a) its battery level is less than 40%, and (b) it cannot reach its destination with the remaining charge. We compare the proposed queue management algorithm against a random allocation of stations where an EV proceeds towards the closest station for recharging. Note that existing algorithms mentioned in Sect. 2 are not immediately available for comparison because these algorithms are not designed to address issues such as (a) trust, (b) monitoring, (c) enforcement, (d) incentive and (e) cost as mentioned in Sects. 1 and 2 (Fig. 2).

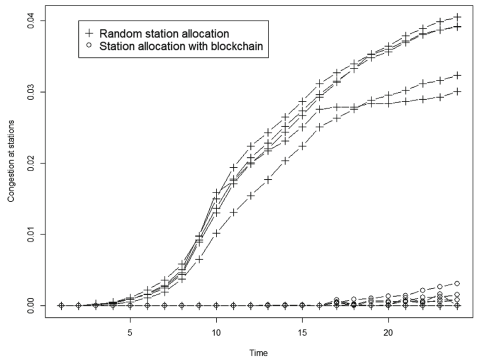


(a) (b)

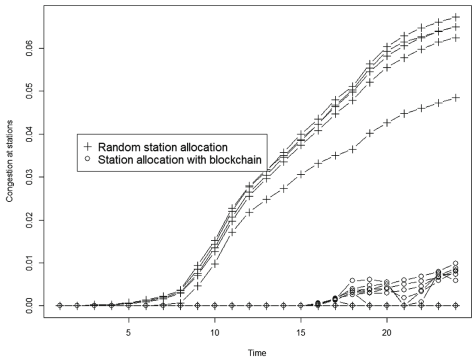
Fig. 2. The EU road network used for evaluation. The figure on the left shows station locations as 'red' nodes and the figure on the right shows station segments as stations with the same colors. (Color figure online)

Figure 3(a, b, c, d) show that the congestion level at stations for the proposed queue management solution performs much better than the random allocation (the congestion for a random station allocation is at least 12 times the congestion for the blockchain-maintained solution). We measure congestion level as the length of queues at the stations.

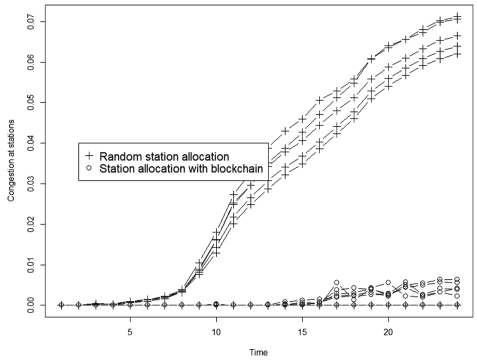
Next, we analyze load balancing among station segments. It is measured as a standard deviation of Score (discussed in Sect. 4.2) for the station segments. We found (Fig. 3(e, f, g, h)) that the standard deviation for our queue management solution is much lower than for random allocation. Hence the proposed solution efficiently load balances the station segments. Finally in Fig. 4, we show that the waiting time for EVs recharging with the proposed queue management solution is at most $\frac{1}{10}$ compared with a random station allocation.



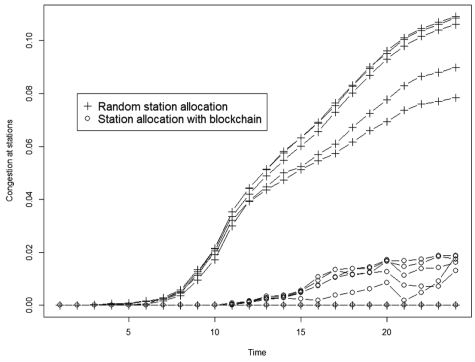
(a) Congestion level for dataset 1



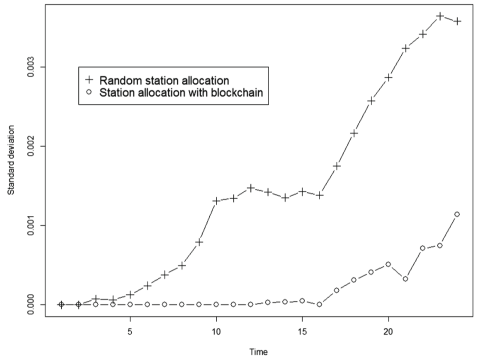
(b) Congestion level for dataset 2



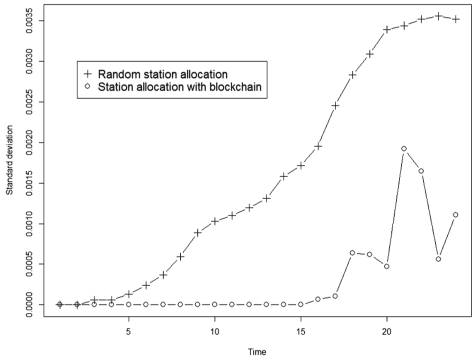
(c) Congestion level for dataset 3



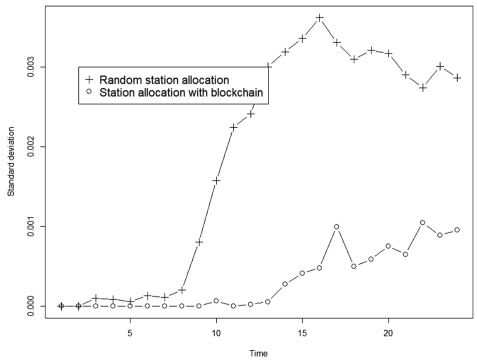
(d) Congestion level for dataset 4



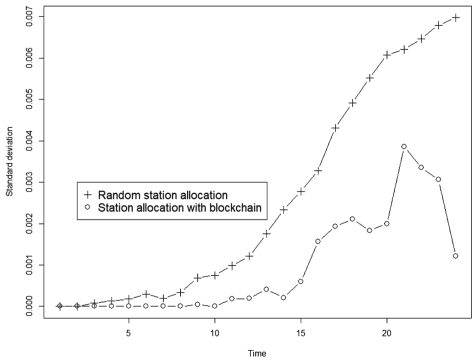
(e) Standard deviation for dataset 1



(f) Standard deviation for dataset 2

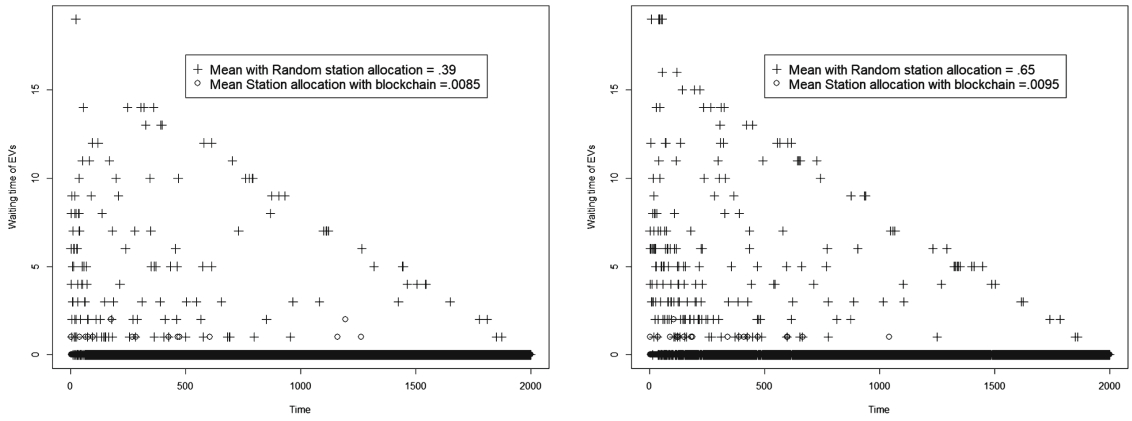


(g) Standard deviation for dataset 3



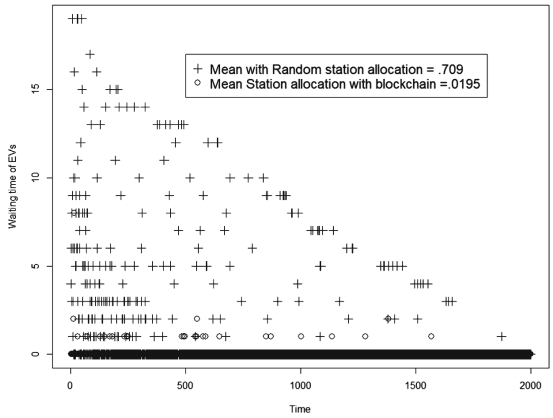
(h) Standard deviation for dataset 4

Fig. 3. Congestion level and standard deviation.

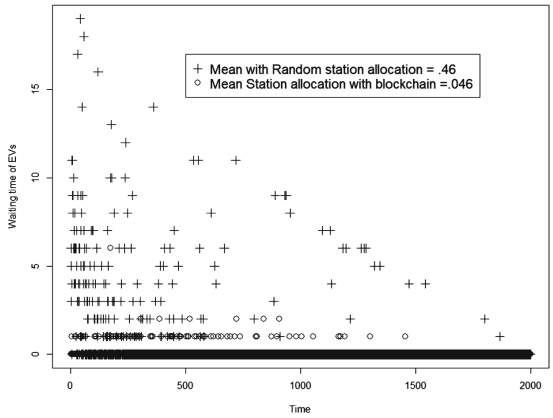


(a) Waiting time for dataset 1

(b) Waiting time for dataset 2



(c) Waiting time for dataset 3



(d) Waiting time for dataset 4

Fig. 4. Waiting time for four datasets

6 Conclusion

In this paper we proposed a blockchain-managed EV queue management procedure. It can support the federation of station networks and allow small businesses to establish charging stations by providing proper monitoring and governance tools. In the future, we will develop a dynamic station segmentation version as an improvement over this solution.

Acknowledgments. This publication has emanated from research supported in part by a research grant from Science Foundation Ireland (SFI) under Grant Numbers SFI/12/RC/2289 and SFI/16/RC/3918, co-funded by the European Regional Development Fund.

References

1. Al-Sultan, S., Al-Doori, M.M., Al-Bayatti, A.H., Zedan, H.: A comprehensive survey on vehicular ad hoc network. J. Netw. Comput. Appl. **37**, 380–392 (2014). https://doi.org/10.1016/j.jnca.2013.02.036
2. Alonso, M., Amaris, H., Germain, J.G., Galan, J.M.: Optimal charging scheduling of electric vehicles in smart grids by heuristic algorithms. Energies **7**(4), 2449–2475 (2014). https://doi.org/10.3390/en7042449. http://www.mdpi.com/1996-1073/7/4/2449
3. King, S., Nadal, S.: PPCoin: peer-to-peer crypto-currency with proof-of-stake (2012). http://www.peercoin.net/assets/paper/peercoin-paper.pdf
4. Nakamoto, S.: Bitcoin: a peer-to-peer electronic cash system (2009). http://www.bitcoin.org/bitcoin.pdf
5. Said, D., Cherkaoui, S., Khoukhi, L.: Multi-priority queuing for electric vehicles charging at public supply stations with price variation. Wirel. Commun. Mob. Comput. **15**(6), 1049–1065 (2015). https://doi.org/10.1002/wcm.2508
6. Shuai, W., Maill, P., Pelov, A.: Charging electric vehicles in the smart city: a survey of economy-driven approaches. IEEE Trans. Intell. Transp. Syst. **17**(8), 2089–2106 (2016). https://doi.org/10.1109/TITS.2016.2519499
7. Thakur, S., Gasperis, G.D.: Cartel formation in charging network for electric vehicles. In: 2016 IEEE 16th International Conference on Environment and Electrical Engineering, EEEIC, pp. 1–6, June 2016. https://doi.org/10.1109/EEEIC.2016.7555829
8. Whaiduzzaman, M., Sookhak, M., Gani, A., Buyya, R.: A survey on vehicular cloud computing. J. Netw. Comput. Appl. **40**, 325–344 (2014). https://doi.org/10.1016/j.jnca.2013.08.004
9. Wu, C., Mohsenian-Rad, H., Huang, J.: Vehicle-to-aggregator interaction game. IEEE Trans. Smart Grid **3**(1), 434–442 (2012). https://doi.org/10.1109/TSG.2011.2166414
10. Xiong, Y., Gan, J., An, B., Miao, C., Soh, Y.C.: Optimal pricing for efficient electric vehicle charging station management. In: Proceedings of the 2016 International Conference on Autonomous Agents & Multiagent Systems, AAMAS 2016, pp. 749–757. International Foundation for Autonomous Agents and Multiagent Systems, Richland (2016). http://dl.acm.org/citation.cfm?id=2937029.2937035
11. Yu, R., Ding, J., Zhong, W., Liu, Y., Xie, S.: PHEV charging and discharging cooperation in V2G networks: a coalition game approach. IEEE Internet Things J. **1**(6), 578–589 (2014). https://doi.org/10.1109/JIOT.2014.2363834