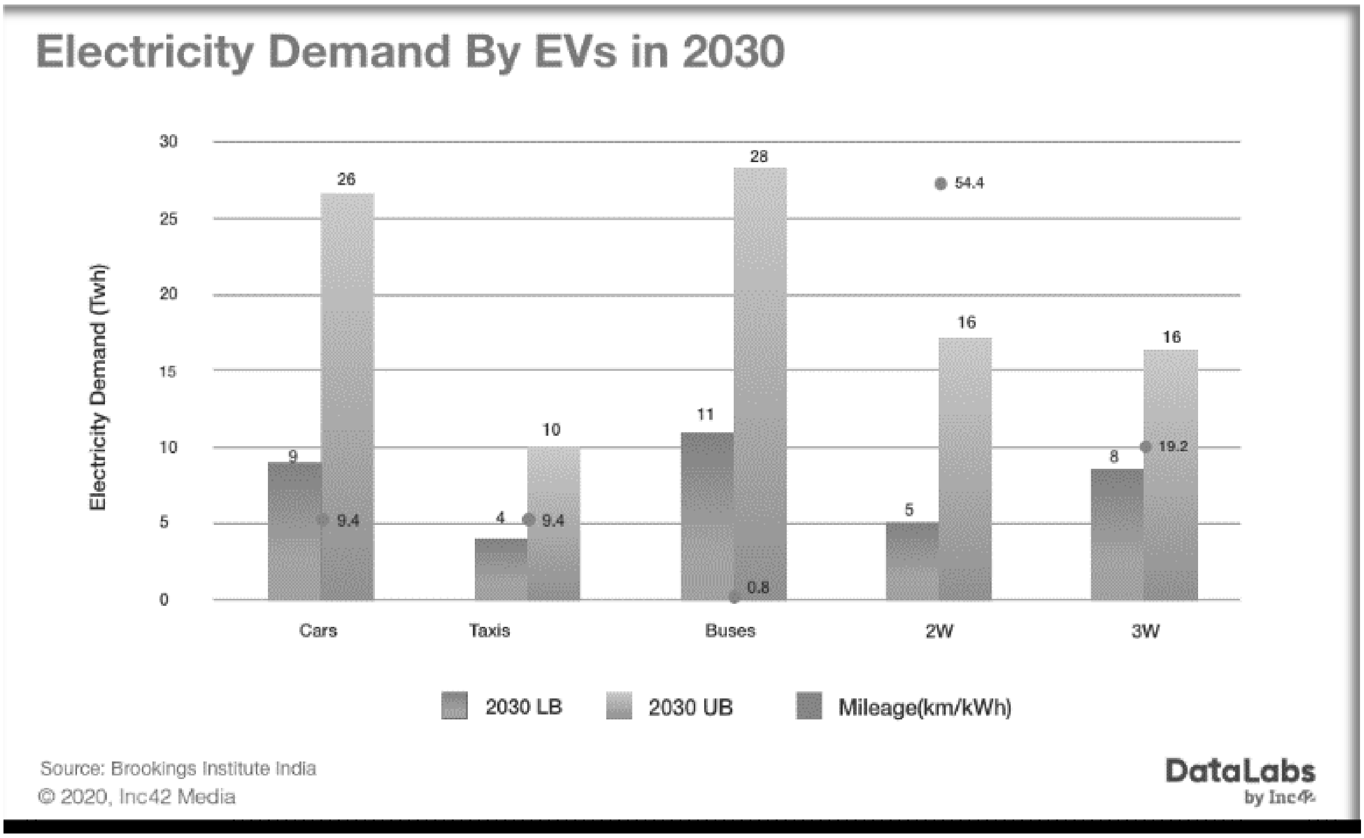
1. Abstract:

Electric Vehicles are rapidly substituting petrol or diesel because of their economical operation and low environmental pollution, However, there still exist obstacles to increase rapidly their use, such as battery charging time, which is longer than traditional oil filling time in cars. This relatively longer charging time can create waiting queues in charging stations in peak hours. At the same time, it is recognized that the current increasing trend of EV use will have a serious impact on the stability of power grids (i.e., electricity providers). The installation of charging stations, which addresses such problems, is essentially required in smart grid communities.

This paper proposes an operational framework for multiple EV charging stations. To reduce the queue size and maintain the power grid stability. Our analysis is based on a multi-queue system, used as a model of charging stations whose dynamics we investigate. Specifically, our interest is the performance change when demand responses (i.e., the behavior of customers) are controlled by varying the charging cost per unit. We proceed with our work in two steps: In the first step, we consider the EV allocation problem. We formulate an optimization problem which can minimize the waiting time of customers and obtain its solution. Then, we additionally regard the size constraint of charging stations and propose an optimal EV allocation algorithm. In the second step, we suggest price control method to get the optimal allocation.

2. Introduction:

The electric vehicle (EV) market in India is expected to hit over 63 lakh unit mark per annum by 2027, according to a report by India Energy Storage Alliance (IESA). In the base



case scenario, the EV market is expected to grow at CAGR of 44 per cent between 2020-2027 and is expected to hit 6.34-million-unit annual sales by 2027.[2]

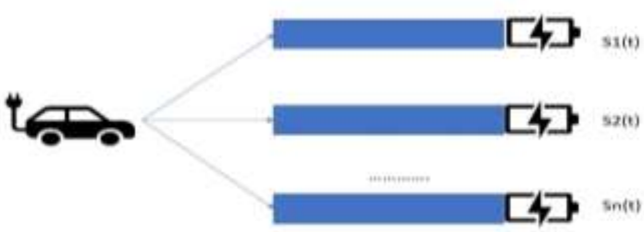
As India gears up to attain 30% market penetration of electric vehicles by 2030, the sales reach for this segment is expected to touch 43 Mn and the stock to grow to more than 250 Mn. The demand for electricity to power EVs is projected to increase to almost 640 TWh by 2030, according to the new policy, and 1,110 TWh in order to meet the EV30@30 goal.

According to DataLabs by Inc42 research [3], India is projected to require about 100 TWh of electricity about 5% of total electricity by 2030 given that 80% of the country's population adopts electric vehicles. So, at this stage, EV charging stations are an essential factor for the success of EVs, along with their performance improvement, they are required to satisfy both the stability of power grids and to offer a reasonable Quality of Service to customers.

We propose an efficient operational framework for multiple charging stations, in the sense of the throughput of stations and the Quality of Service offered to customers, and also decreasing the market monopoly of one station. the behavior of customers has a high impact on the performance. In general, charging stations provide different service levels due to their facility size (e.g., the number of charging plugs or waiting spaces). Also, the charging speed of plugs may vary. As a result, the customer behavior affects not only their QoS (e.g., expected time to charge), but also the throughput of the stations. Under these circumstances, our objective is to introduce a simple platform for customers to induce QoS maximization behavior. This platform will show the real time charging price of a charging station and their status (e.g., fully occupied or not) to the customer so they can decide whether they want to charge their EV at that time or not (expected that higher price will demotivate EV owners to charge). In this way we can control the customer behavior.

3. Problem Description:

We consider an urban area, where multiple charging stations exist in the neighborhood, and define **N** as the station set, where **|N |=n**, number of charging stations. Each station corresponds to a single queue structure as shown in following Fig. (i.e., n queue system). We set $S_{i(t)}$ as the size of queue **i** at time **t**, where $i \in N$.



For customers, the charging takes an exponential amount of time with rate μ_i, where **i = {1, . . ., n}**. Further, we consider that the occurrence of EV charging events follows a Poisson process with rate λ. From the assumption that n stations are located in the neighborhood and by Poisson thinning property, the arrival process at each station is also Poisson with rate λ_i. Under these conditions, each queue in the system is a **M/M/1** queue and it suffices to analyze their queue dynamics by considering their input and output rates.

For convenience, we define vectors $\vec{\lambda}$ and $\vec{\mu}$ whose elements denote the rate of EV arrival **($\vec{\lambda}$)** and station service **($\vec{\mu}$)**, respectively. Customers enter one of the stations after the charging station selection and this lets us consider a constraint such that $\sum_{i=1}^{n} \lambda_i$ is equal to **λ**.

- **Vehicle Allocation Analysis:**

Our first objective is to study a load balancing policy to optimize vehicle allocation. This includes both minimizing the waiting time of customers and preventing the overflow of a station's waiting size.

A. Optimized Load Balancing:

From the stationary distribution of **M/M/1** queue, the mean queue size **(E[N])** at each station is computed by **E[Ni]=** $\frac{u_i}{u_i-\lambda_i}$ .This is directly related to the waiting time of customers in a queue. By Little's law, **E[Ni]** is proportional to the waiting time. For this reason, we consider the following optimization problem which minimizes the global queue size.

$$\begin{aligned} \underset{\vec{\lambda}}{\text{minimize}} \quad & \sum_{i=1}^{n} \frac{\mu_i}{\mu_i - \lambda_i} \\ \text{subject to} \quad & \sum_{i=1}^{n} \lambda_i = \lambda \\ & \mu_i - \lambda_i > 0 \\ & \mu_i > 0,\ \lambda_i > 0 \end{aligned}$$

This finds the vector $\vec{\lambda}$ which minimizes the global number of waiting customers. The value λ is the occurrence rate of EV charging events and each subjective condition describes the load balancing through Poisson thinning, stability, positive arrival and departure rates, respectively.
Daehyun Ban *et.al.* [1] have solved this in their work, as shown by them
The solution vector $\vec{\lambda}$ of this problem is given by:

$$\lambda_i^{\text{opt}} = \frac{\lambda + \sum_{j=1}^{n} (\sqrt{\mu_i \mu_j} - \mu_j)}{\sum_{j=1}^{n} \sqrt{\frac{\mu_j}{\mu_i}}}, \text{ where } i = \{1, \ldots, n\}$$

B. Station Size Limitation and Vehicle Allocation Algorithm:

The optimal allocation requires a stability condition on local queues. During peak times, the incoming PHEV rate can be temporarily higher than service rate (i.e., $\lambda opt_i > \mu_i$) and this leads to an increase of the local queue size.

To mitigate this temporal station capacity overflow, we utilize a size-based active queue management (AQM). We set a congestion indicator ci(t) as shown below:

$ci(t) = 1_{\{Qi(t)>Li\}}$, where $i \in N$, Li indicates the size capacity of station i.

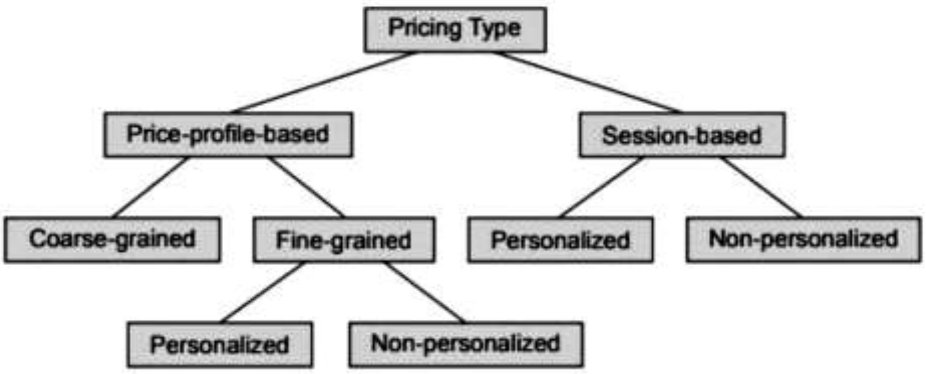
Algorithm 1 PHEV Allocation Control by using AQM

Require: λ, $\vec{\mu}(t)$ and $\vec{c}(t)$ with size $|\mathcal{N}|$
  for $i = 1 \to |\mathcal{N}|$ **do**
    if $c_i(t) = 1$ **then**
      $\mu_i(t) \leftarrow 0$ (Exclude Station i)
    end if
  end for
  $\overrightarrow{\lambda^{\mathrm{opt}}}(t+1) \leftarrow \frac{\lambda + \sum_{j=1}^{|\mathcal{N}|}(\sqrt{\mu_i \mu_j} - \mu_j)}{\sum_{j=1}^{|\mathcal{N}|}\sqrt{\frac{\mu_j}{\mu_i}}}$

By setting $\mu_i = 0$ for the station i experiencing congestion, Alg. 1 induces $\lambda_i^{opt} = 0$.

- **Customer Control by Using Dynamic Pricing:**

Steffen Limmer [4] has shown that the pricing strategies can be categorized according to the following figure.



Price-profile-based pricing sets different prices (usually per energy unit) for different time intervals. Mostly used Price profile-based are fine-grained price profiles, which set an individual price for each small scheduling interval which may be for 5 minutes or one hour. However, some authors like Guo *et al*. [5] propose coarse-grained price profiles, which set a constant charging price for a longer time.

For our work we will try to set the price for hourly time slots, so our strategy is a fine-grained pricing strategy.

We considered the price update framework as a discrete time system. Each timeslot duration is T and stations have the arrival rate λ_i and service rate μ_i in that time slot for all $i \in N$.

Now we proceed with the price control of stations by using the arrival rate difference between the optimal λ_i^{opt} (t) and $\lambda_i(t)$. At time t, we set this difference to di(t):

$$d_i(t) = \lambda_i(t) - \lambda_i^{opt}(t), \text{ where } i \in N$$

a positive di(t) means that more vehicles than optimal are entering into station i, meaning that it is required to decrease the input rate.

In addition, the opposite case also holds. By controlling the price of stations, the price for the station *i* at tie slot *t* will be:

$$p_i(t) = \text{Base price} + \gamma d_i(t), \text{ where } i \in N$$

γ is a constant.

Now we have to know about the arrival rate of the vehicles at time slot *t*.

- **Predicting the arrival rate:**

Let's say we want to set the price for the time slot 10 a.m. to 11 a.m. for doing so we need to know how many vehicles may come in that time slot. As we don't know what will be the arrival rate for that slot, we need to predict it. The dataset we used for predicting the arrival rate is taken from 'Geeksforgeeks'.[8]. The dataset has two columns which are useful to us one is hourly time slot and another is number of vehicles that have arrived in that time slot.

```
dataset.head()
```

	DateTime	Junction	Vehicles	ID
0	1/1/2016 0:00	3	3	20160101003
1	1/1/2016 1:00	3	5	20160101013
2	1/1/2016 2:00	3	4	20160101023
3	1/1/2016 3:00	3	1	20160101033
4	1/1/2016 4:00	3	1	20160101043

As a step of preparing the dataset, we extracted the following features from the dataset.

	DateTime	Vehicles	date	weekday	hour	month	year	dayofyear
0	2016-01-01 00:00:00	3	1	4	0	1	2016	1
1	2016-01-01 01:00:00	5	1	4	1	1	2016	1
2	2016-01-01 02:00:00	4	1	4	2	1	2016	1
3	2016-01-01 03:00:00	1	1	4	3	1	2016	1
4	2016-01-01 04:00:00	1	1	4	4	1	2016	1

Now, we have used random forest regression technique on the datasets for predicting the arrival rate.

a. *Random forest regression:*

Random Forest is a popular machine learning algorithm that belongs to the supervised learning technique. It can be used for both Classification and Regression problems in ML. It is based on the concept of ensemble learning, which is a process of combining multiple classifiers to solve a complex problem and to improve the performance of the model. As the name suggests, "Random Forest is a classifier that contains a number of decision trees on various subsets of the given dataset and takes the average to improve the predictive accuracy of that dataset." Instead of relying on one decision tree, the random forest takes the prediction from each tree and based on the majority votes of predictions, and it predicts the final output.

b. *Why use Random Forest?*

It takes less training time as compared to other algorithms.
It predicts output with high accuracy, even for the large dataset it runs efficiently.
It can also maintain accuracy when a large proportion of data is missing.

- **Accuracy of the models:**

Station1:

```
R squared: 97.62
Mean Absolute Error: 3.008327010622155
Mean Square Error: 17.1778998861912
Root Mean Square Error: 4.144623008934733
```

Station2:

```
R squared: 97.70
Mean Absolute Error: 2.9087708649468893
Mean Square Error: 16.243958497723824
Root Mean Square Error: 4.030379448355182
```

Station3:

```
R squared: 97.59
Mean Absolute Error: 3.0443740515933233
Mean Square Error: 17.251058459787558
Root Mean Square Error: 4.153439353088902
```

Stattion4:

```
R squared: 89.51
Mean Absolute Error: 1.7594916540212442
Mean Square Error: 4.862227617602429
Root Mean Square Error: 2.2050459445559016
```

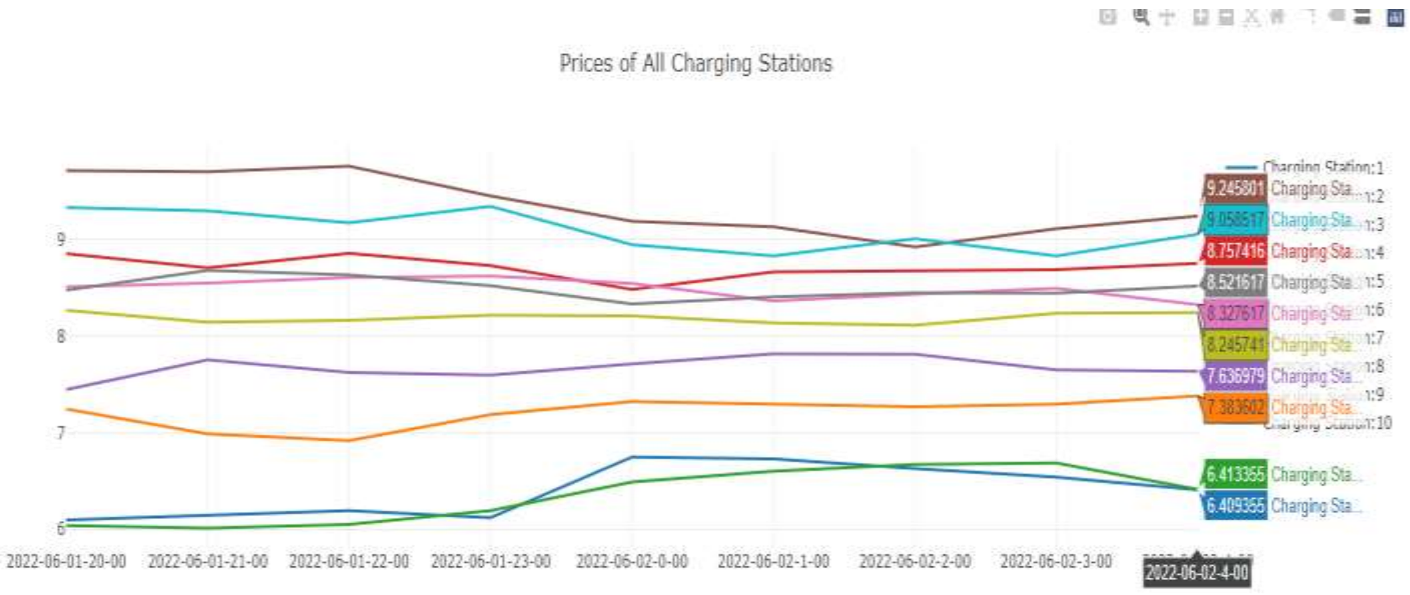
Station5:

```
R squared: 95.41
Mean Absolute Error: 2.2212193251533736
Mean Square Error: 7.951061426380367
Root Mean Square Error: 2.8197626542637177
```

Station6:

```
R squared: 88.64
Mean Absolute Error: 1.81675
Mean Square Error: 5.329748409090909
Root Mean Square Error: 2.308624787420188
```

4. Results:



According to a report by INC42 [10] on an average the EV's take up to 30 unit of electricity. So, for the traditional fixed pricing strategy (8 rupees per unit), the income of the 10 charging stations for one day according to the dataset would look like following:

Station 1	Station 2	Station 3	Station 4	Station 5	Station 6	Station 7	Station 8	Station 9	Station 10
19740	10080	8610	14700	19005	14805	15645	10605	14805	13650

*Prices are in rupees

We can clearly see that station 1 and 5 is having duopoly, where station 3 is having very less income. After implementing the strategy, the following figure shows the pricing for a time slot in different charging stations. The base price was taken as 8 rupees as this is the average price currently in Indian charging stations.

The higher price simply indicates that, the charging station is getting more number arrivals of the EVs than the other stations.

The objective of our dynamic pricing is to achieve the optimal arrival rate. So, if we can achieve the optimal rate and instead of using the fixed pricing if we take the dynamic prices the income on the same day looks like following:

Station 1	Station 2	Station 3	Station 4	Station 5	Station 6	Station 7	Station 8	Station 9	Station1 0
16329	14164	15010	13115	16017	14239	13766	14050	15726	13119

*Prices are in rupees

So, we can see that the income has been distributed among the charging stations, and there is no market monopoly.