DQ-Map: Dynamic Decision Query Mapping for Provisioning Safety-as-a-Service in IoT

Chandana Roy*, *Student Member, IEEE*, Chandrani Roy Chowdhury§, Sudip Misra†, *Senior Member, IEEE*, and Jhareswar Maiti‡, *Member, IEEE*,
*‡Department of Industrial and Systems Engineering, §†Department of Computer Science and Engineering,
*§†‡Indian Institute of Technology Kharagpur, India,
Email: {*chandanaroy, †sudipm}@iitkgp.ac.in, ‡jmaiti@iem.iitkgp.ernet.in, §crc24jan@gmail.com

***Abstract*—In this work, we propose a dynamic decision query mapping mechanism, DQ-Map, for provisioning Safety-as-a-Service (Safe-aaS) [1]. A Safe-aaS infrastructure provides customized safety-related decisions simultaneously to multiple end-users. We consider road transportation as the application scenario of Safe-aaS and termed the safety-related decision to be delivered to the end-users as *decision queries* (DQs). These DQs are generated according to the decision parameters selected by the end-users. The primary aim of our proposed work is to reduce the total number of sensor nodes required to generate the safety-related decisions, which minimizes both energy and time consumption. Further, the requested DQs are processed and decision is generated in three different stages. First, the DQs are categorized as *emergency* (EDQ) and *non-emergency* (NEDQ), depending upon the type of vehicle from where the end-users have requested for safety services. The EDQs and NEDQs are mapped with the stored decisions present in the database of the decision virtualization layer during the second level. In case of mismatch with the stored decisions in the database, EDQs are directly executed from the sensor nodes deployed at a particular geographical location or into the vehicles, in the device layer of the Safe-aaS infrastructure. In the third level, the similarity score of NEDQs, which do not match with the parameters of the stored decisions, are computed. Based on the number of similar decision parameters present in them, the similarity score is computed. Extensive simulation results of the proposed scheme, DQ-Map, depicts that the amount of energy consumed and time required to generate a decision is reduced by 55.16% and 54.55% respectively, compared to the traditional Safe-aaS architecture.**

***Index Terms*—Road transportation, Decision Mapping, Safety-as-a-Service, Decision Query, Safety, IoT.**

I. Introduction

THE integration of Internet of Things (IoT)-based technologies to be applicable across different industrial sectors have resulted in the automation of the industrial processes [2]. Internet of Vehicles (IoV) [3], a domain of Industrial Internet of Things (IIoT), help in controlling traffic and road congestion. Further, certain applications of IoV such as Intelligent Transportation System (ITS) [4] and Advanced Driver Assistance Systems (ADAS) [5] help in the reduction of on-road congestion, pedestrian detection, and assist drivers. However, the rate of road accidents and casualty have increased significantly with the increase in the number of on-road vehicles. Human errors such as drowsiness, driving behavior, and different socio-economic reasons, act as one of the key factors in the occurrence of the road accidents [6]. However, prior delivery of on-road safety-related informations help to significantly reduce the human errors and other associated factors which lead to an accident.

In this paper, we propose a dynamic decision query mapping mechanism, *DQ-Map*, for provisioning Safety-as-a-Service [1]. A Safe-aaS architecture provides customized safety-related decisions simultaneously to multiple end-users, based on the decision parameters selected by them. Considering road transportation as the application scenario, we define the decision to be delivered to the end-users, as per the decision parameters chosen by them, as *Decision Queries* (DQs). Our primary focus is to minimize the total number of sensor access for the generation of safety-related decisions. In order to reduce these number of sensor access, we map the parameters of DQs with the parameters of the stored decisions.

With the increase in the number of on-road vehicles, the drivers and organizations adopted different safety measures. A Safe-aaS infrastructure provides customized safety-related decisions simultaneously to the multiple registered end-users, founded on the concept of decision virtualization. Typically, in road transportation, the timely delivery of the requested decision parameters to the end-users is essential to avoid congestion and accidents. Based on the decision parameters selected by the end-users, the decision is provided to them. However, in a practical scenario, some of the decision parameters such as number of turns, manholes, and potholes on the road selected by the end-users, may not vary frequently. Therefore, multiple time execution of the same parameters result in unnecessary energy and time consumption. On the other hand, the sensor nodes are energy-constrained in nature. This strongly motivated us to propose a scheme for dynamic mapping of the decision parameters requested by the end-users with the parameters of the stored decisions present in the database. The decision to be delivered to these end-users are termed as *decision queries*.

In this work, we consider the timeliness of the delivery of decisions in Safe-aaS, based on the type of decisions requested. The proposed problem provides solution to the following: (a) categorization of requested decision parameters and accordingly serve these requests, (b) dynamic mapping of decision parameters with the parameters of the stored decisions, and (c) find the optimal number of sensor nodes required to generate a decision. The primary contribution of this work is to propose the mechanism to dynamically map the decision parameters requested by the end-users with the decisions generated. The specific *contributions* are as follows:

- We propose a three-level decision query mapping mechanism in the decision virtualization layer of a Safe-aaS architecture. We formulate a integer linear programming (ILP) to compute the minimum number of sensor nodes required to generate a decision. Further, we solved the ILP applying Karush-Kuhn-Tucker (KKT) conditions.
- We classify the safety-related requested DQs into emergency (EDQ) and non-emergency decision query (NEDQ). The decision parameters present in the EDQs and NEDQs are compared with the stored decisions. In case of mismatch, EDQs are directly executed from the sensor network. NEDQs are executed through the third level of decision generation.
- We propose three algorithms for the decision query mapping procedure – DQ-Class, DQ-Map, and DQ-Sim. Extensive analysis of our proposed scheme, DQ-Map, depicts that the energy consumption and decision generation time reduces significantly compared to the existing Safe-aaS architecture, AEB, ACPS, and OBFBT.

II. Related Work

In this section, we discuss the prior research works done in the domain of road transportation. We discuss the existing research works depending upon their application scenario such as heavy and automated vehicles [7], [8], analysis of reasons leading to accidents [9]–[13], and infrastructures and systems developed to improve road safety [1], [14]–[20].

Liang *et al.* [7] proposed an optimized solution to improve the fuel efficiency of heavy vehicles, by forming platoons with the neighboring vehicles. In the proposed work, the authors designed an algorithm such that the road topography has minimum influence on the vehicles of the same route, when organized as platoons. Further, as the movement of automated vehicles can be controlled and co-ordinated, therefore, this results in minimizing the on-road congestion. Meissner *et al.* [8] proposed a coordination-based algorithm, which maximizes the number of successful exits in a highway and uniformly distributes the traffic.

There are various causes such as human errors, weather conditions, and road conditions. Caban *et al.* [9] studied the various reasons of accidents, the different type of injuries related with it, and the engineering materials required for the treatment. One of the major reasons for accidents is drowsiness of the drivers during driving. Chui *et al.* [10] developed an accurate driver drowsiness classifier using electrocardiogram genetic algorithm-based support vector machine. The authors proposed three classifiers – classifier 0, 1, and 2, to detect the different state of drowsiness of the drivers. In another work, Dey *et al.* [11] collected the road weather data from different sources and suggested intelligent transportation system-based solutions to minimize the impact of adverse route-specific weather conditions.

On the other hand, to maintain safe speed and distance with the other vehicles, Bertolazzi *et al.* [16] designed a driver support system, based on the lane curvatures. Similarly, Amditis *et al.* [17] proposed a safety framework, called INSAFES, which worked in three levels – perception, decision, and action. The proposed framework provides information of safe distance, lane change, safe speed, and collision avoidance. Further, Roy *et al.* [1], [14], [15] proposed an infrastructure, Safe-aaS, to provide customized safety-related decisions to the multiple registered end-users simultaneously. Additionally, authors considered the presence of heterogeneous type of static and mobile sensor nodes. Mobile sensor nodes are deployed into the vehicles, while static sensor nodes are deployed at a particular geographical location. In a Safe-aaS architecture, the time-critical data are primarily processed at the edge nodes. With the mobility of the vehicles, the edge nodes present within their vicinity changes. Therefore, appropriate selection of edge node is essential. On the other hand, certain privacy-related problems exist in the Safe-aaS architecture. Considering these privacy issues, the authors proposed a blockchain-enabled Safe-aaS architecture to be applicable across different industrial verticals. Further, the service region of any Safety Service Provider (SSP) is bounded, which results in the interruption of safety services provided to the end-users. Roy and Misra [19] proposed a service handoff scheme to provide uninterrupted services to the end-users.

Synthesis: Different research works explore various reasons of accidents, on-road safety issues, and improvement of fuel efficiency in the vehicles. In the existing literature [16], [17], the authors proposed an infrastructure to integrate the various safety aspects such as safe distance, safe speed, and collision avoidance. Roy *et al.* [1] proposed the theoretical model and analyzed the cash inflow and outflow among the various actors of the Safe-aaS infrastructure to provide safety-related decisions to the end-users. However, the dynamic mapping of the end-users' requests with the decisions stored in the database present in the decision virtualization layer, is not yet addressed. This mapping of the requested decision parameters with the parameters of the stored decisions result in the reduction of energy and time consumption. Therefore, we propose the decision query mapping mechanism, prioritizing the emergency decision requests.

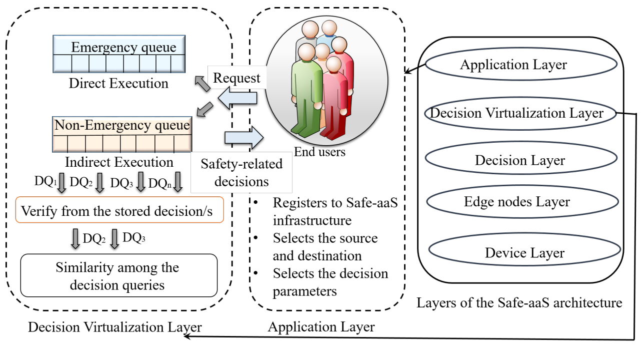
Emergency queue
Direct Execution
Request
Non-Emergency queue
Indirect Execution
Safety-related decisions
End users
DQ1 DQ2 DQ3 DQn
Verify from the stored decision/s
DQ2 DQ3
Similarity among the decision queries
Registers to Safe-aaS infrastructure
Selects the source and destination
Selects the decision parameters
Application Layer
Decision Virtualization Layer
Decision Layer
Edge nodes Layer
Device Layer
Layers of the Safe-aaS architecture
Decision Virtualization Layer
Application Layer

Fig. 1: Decision Virtualization Mapping Framework

III. Problem Description

A. Problem Scenario

In the proposed work, we consider an ITS scenario where *Safe-aaS* [1] architecture is implemented. A Safe-aaS infrastructure comprises five layers – *device*, *edge*, *decision*, *decision virtualization*, and *application*. The device layer consists of heterogeneous types of static and mobile sensor nodes, which sense and transmit data to the edge layer/cloud, based on the time-criticality of data. Static sensor nodes are deployed at a particular geographical location, while mobile sensor nodes are deployed into the vehicles. The primarily processed sensor data are transferred to the decision layer for further processing. Thereafter, the multiple processed sensor data are combined to generate the decision requested by end-users in this layer. Further, the logical mapping between the end-users requests and decision generated is done in the decision virtualization layer. The four principal actors of *Safe-aaS* architecture are – vehicle owners, sensor owners, safety service provider (SSP), and end-users. The application layer acts as the interface between the end-users and Safe-aaS. The end-users register to this architecture through a Web portal and provides their source and destination. Each of the registered end-users select certain decision parameters such as location and depth of potholes, road congestion, and weather conditions, among the available ones, to the *Safe-aaS* infrastructure, as demonstrated in Fig. 1. Based on the decision parameters selected by them, the decisions are delivered to the end-users. The end-users enjoy these services on pay-per-use basis. However, the end-user remains completely unaware of the back-end process of decision generation.

The primary objective of our proposed work is to reduce the sensor access frequency, total query execution time, and maintain the sustainability of the underlying sensor network. We term the safety-related decisions to be delivered to the end-users, as per the decision parameters selected by them as *Decision Query* (DQ). We propose a dynamic *Decision Query Mapping* mechanism in the decision virtualization layer of Safe-aaS architecture. Fig. 1 illustrates the different stages of decision query mapping in the *Safe-aaS* infrastructure. We map the parameters of DQs requested by the end-users with the decisions stored in the database. In the first level, the DQs are categorized based on the Emergency (EDQ) and Non-Emergency (NEDQ) end-users request. Each of the DQs are associated with the type of vehicle from where the decisions are requested. We categorize the DQs requested from emergency-type vehicles such as ambulance and VIP's vehicle as EDQ. The other requested DQs are categorized as NEDQ. Both EDQs and NEDQs are passed through the second level, which addresses the mapping of DQs with the already generated decisions stored in the database. The EDQs, which does not matches, are directly executed from the underlying sensor network. In case of mismatch, NEDQs are only forwarded to the third level, which analyzes the overlap issues/similarity of decision parameters among the NEDQs.

B. Problem Formulation

We consider that n number of registered end-users are present in the Safe-aaS architecture at a particular time instant, t. A set of registered end-users is represented as $\mathbb{EU}^t = \{EU_1^t, EU_2^t, \cdots, EU_n^t\}$, where EU_i^t is denoted as the i^{th} end-user at time-instant, t. We assume that each of the registered end-user requests for a DQ, DQ_i^t, at the t^{th} time instant. Further, a set of the number of DQs requested by the end-users at the time instant, t, is represented as, $\mathbb{D}^t = \{DQ_1^t, DQ_2^t, \cdots, DQ_n^t\}$. Any of the i^{th} registered end-users' requests for safety-related decisions from his/er current *location*. However, with the mobility of the end-user, the location of the requested decision parameters changes, which results in the variation of the generated decision query, DQ_i^t. Therefore, the present *location* of DQ_i^t associated with the i^{th} end-user is represented as: $loc_i^t = \{lat_i^t, lon_i^t\}$, where lat_i^t and lon_i^t denote the latitude and longitude of the i^{th} end-user, respectively. The requested decision query, DQ_i^t comprises multiple decision parameters, Dp_i, present location of the end-user, loc_i^t, target application area, tr_i, for which the service is requested, and time stamp, ts_i. Therefore, DQ_i^t is expressed as a four-tuple, $DQ_i^t = \langle Dp_i^j, loc_i^t, tr_i^t, ts_i \rangle$ where $j \in c$ and c denotes the total number of decision parameters available in the Safe-aaS architecture. Dp_i represents the decision parameters selected by the i^{th} end-user.

C. Preprocessing Stage

Let us consider $\mathbb{DB}$ as the database with m number of generated decisions at any time instant t, represented as $\mathbb{DB}^t = \{D_1^t, D_2^t, \cdots, D_m^t\}$. We assume that the total number of decision parameters present in the i^{th} stored decision, D_i^t is denoted as dpc_i. Therefore, the total number of decision parameters present in the stored decisions at any time instant, t, is represented as N_t. Mathematically:

$$N_t = \sum_{i=1}^{m} dpc_i \tag{1}$$

We assume that a hash table, H, keeps record of the count and type of decision parameters present in the generated decisions stored in the database, at any time instant t. The hash key of the table, H represent the number of decision parameters present in a decision.

***Proposition* 1.** *The total number of hash keys stored and the maximum number of columns present in the hash table, H, at any time instant, t must not exceed c and m, respectively.*

Proof: We consider that the total number of decision parameters served by the *Safe-aaS* architecture is denoted as c. Further, each of the i^{th} hash keys stored in the hash table, H, comprises of the number of decision parameters contained in the i^{th} decision, D_i. Hence, at any time instant, t, the maximum number of decision parameters present in any of the stored decisions does not exceeds c. Therefore, it is proved that the total number of hash keys in H at the t^{th} time instant is c.

On the other hand, the total number of decisions stored in the database is assumed as m. The number of entries in any row of the hash table, H vary with time. However, the total number of columns does not exceed m, at any time instant t. Mathematically,

$$m = \sum_{i=1}^{c} dp_i, \tag{2}$$

where dp_i is the number of decisions present in the i^{th} row of the hash table, H. ■

In order to visualize the similarity between the two generated decisions from the database in $O(1)$ time, we construct a decision similarity matrix, $\mathbb{M}_{\mathbb{DB}^t}$, of dimension $(m \times m)$. Each entry in the $\mathbb{M}_{\mathbb{DB}^t}$ denotes the number of common decision parameters between the i^{th} and j^{th} decision. In other words, each of the element in the decision matrix represents the *similarity score among the stored decisions* (SD_{ij}). The $\mathbb{M}_{\mathbb{DB}^t}$ is symmetric in nature, which is represented as,

$$\mathbb{M}_{\mathbb{DB}^t} = \begin{pmatrix} SD_{11} & SD_{12} & \cdots & SD_{1m} \\ SD_{21} & SD_{22} & \cdots & SD_{2m} \\ \vdots & \vdots & \ddots & \vdots \\ SD_{m1} & SD_{m2} & \cdots & SD_{mm} \end{pmatrix} \tag{3}$$

Therefore, the size of the search space, s is represented as,

$$s = \frac{m \times (m+1)}{2} \tag{4}$$

In order to match the requested DQs with the decisions stored in the decision database, the similarity score, SD_{ij}, is computed. Further, the decision matrix, $\mathbb{M}_{\mathbb{DB}^t}$ provides the similarity score among the stored decisions in $O(1)$ time. In case of mismatch, the NEDQs are forwarded to the third level.

***Definition* 1.** *Similarity Score of Non-Emergency Decision Query (SC_{ij}): The similarity score between any two NEDQs, i and j, is defined as the number of similar decision parameters existing in the i^{th} and j^{th} NEDQ. Mathematically,*

$$SC_{ij} = \mid sp_{ij} \mid \tag{5}$$

where sp_{ij} is the number of similar decision parameters existing in the i^{th} and j^{th} decision query. In order to keep track of the similarity score among the NEDQs, the SC_{ij} for each NEDQs are stored in the *Decision Query Similarity matrix*, $\mathbb{M}^t_{DQ}$. Further, the matrix $\mathbb{M}^t_{DQ}$ is of the order $(n \times n)$ and is symmetric in nature.

$$\mathbb{M}^t_{DQ} = \begin{pmatrix} SC_{11} & SC_{12} & \cdots & SC_{1n} \\ SC_{21} & SC_{22} & \cdots & SC_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ SC_{n1} & SC_{n2} & \cdots & SC_{nn} \end{pmatrix} \tag{6}$$

In the second level of our proposed approach, we check whether the decision parameters requested by the end-users match with the parameters of the decisions stored in the database. In case of mismatch, we check the similarity among the requested DQs. Further, the mismatched NEDQs are generated from the sensor data. The primary aim of this proposed work is to minimize the total number of sensor access. Further, the total number of sensor utilized for the generation of n number of decision queries at any time instant, t, is denoted as $\mathbb{TA}^t$, such that $\mathbb{TA}^t = \sum_{i=1}^{n} TA^t_i$. We represent TA^t_i as a function of DQ^t_i and SC_{ij} for the i^{th} DQ, such that, $TA^t_i = f(DQ^t_i, SC_{ij})$. Therefore,

$$\mathbb{TA}^t = \sum_{i=1}^{n} (f(DQ^t_i, SC_{ij})) \tag{7}$$

On the other hand, each DQ^t_i is a function of the number of decision parameters of the i^{th} DQ, executed from the database and the sensor network respectively. $NDQ^{i,t}_{dv}$ and $NDQ^{i,t}_{sn}$ represent the number of decision parameters executed from the database and sensor network respectively. Thus, $DQ^t_i = f(NDQ^{i,t}_{dv}, NDQ^{i,t}_{sn})$, and Equation 7 is represented as,

$$\mathbb{TA}^t = \sum_{i=1}^{n} \left(NDQ^{i,t}_{dv} + NDQ^{i,t}_{sn} + SC_{ij}\right) \tag{8}$$

Therefore, the ILP is formulated as,

$$\underset{NDQ^{i,t}_{sn}}{argmin} \quad \mathbb{TA}_t \tag{9}$$

subject to,

$$\mathbb{NDQ}^t_{dv} > \mathbb{NDQ}^t_{sn} \text{ and } SC_{ij} \leq SC^{max}_{ij} \tag{10}$$

where $\mathbb{NDQ}^t_{dv}$ and $\mathbb{NDQ}^t_{sn}$ denote the total number of decision parameters executed from the database and sensor network respectively. Further, $\mathbb{NDQ}^t_{dv} = \sum_{i=1}^{n} NDQ^{i,t}_{dv}$ and $\mathbb{NDQ}^t_{sn} = \sum_{i=1}^{n} NDQ^{i,t}_{sn}$. SC^{max}_{ij} represents the maximum number of decision parameters of the i^{th} decision query, which match with the j^{th} decision query. To convert the ILP into simplified form, we apply *Lagrangian* function for Equation (9), which is represented as:

$$\mathcal{L}_t = \sum_{i=1}^{n} (f(DQ^t_i, SC_{ij})) + \lambda_1 \sum_{i=1}^{n} (NDQ^{i,t}_{dv} - NDQ^{i,t}_{sn}) + \lambda_2 \sum_{i=1}^{n} (SC^{max}_{ij} - SC_{ij}) \tag{11}$$

where λ_1 and λ_2 are the *Lagrangian* multipliers. In order to find the optimum solution of the Lagrangian function, we apply *Karush-Kuhn-Tucker* (KKT) [21] conditions. Among the DQs requested, the number of DQs executed directly from the sensor network varies continuously with time, therefore the function is differentiable for number of DQs executed from the sensor nodes. Hence, the local optimum solution is obtained at that point. The *dual feasibility* and *complementary slackness* conditions are represented by Equation (12).

$$\partial_{NDQ^{i,t}_{sn}} \mathcal{L}_t = (1 - \lambda_1) + \sum_{j=1, j \neq i}^{(n-1)} NDQ^{j,t}_{sn} (1 + \lambda_1) = 0 \tag{12a}$$

$$\lambda_i \geq 0 \text{ and } \lambda_i X_i = 0, \forall i = \{1, 2\} \tag{12b}$$

where X_i denote the constraints of Equation (9). Therefore, the minimum number of sensor access at the t^{th} time instant is computed from the KKT conditions.

IV. Solution Approach

We propose a set-intersection-based algorithm to represent the mapping of decision queries. We assume that there are n number of decision queries requested by n users at any time instant t. Further, we consider that the decision database stores m decisions. The similarity among these stored decisions are retained in the matrix, $\mathbb{M}_{\mathbb{DB}^t}$. Algorithm 1 provides insight towards the task of categorizing the DQs into EDQs and NEDQs. The EDQs are first matched with the existing decisions. Thereafter, in case of mismatch, EDQs are executed directly from the underlying sensor network, without pre-processing them in the decision virtualization layer.

Algorithm 1 DQ-Class

```
INPUTS: $DQ$, EDQ, NEDQ
OUTPUTS: $ndq_{dv}$, $ndq_{sn}$
PROCEDURE:
1: for $i = 1$ to $n$ do
2:     if $DQ_i.type = EDQ$ then
3:         $EDQ.enqueue(DQ_i)$
4:         $ndq_{sn} \leftarrow ndq_{sn} + 1$
5:     else
6:         $NEDQ.enqueue(DQ_i)$
7:         $ndq_{dv} \leftarrow ndq_{dv} + 1$
8:     end if
9: end for
```

Algorithm 2 DQ-Map

```
INPUTS: $DQ$, $\mathbb{DB}$, $H$, $\mathbb{M}$, $NEDQ$, $EDQ$
OUTPUTS: $ndq_{dv}$, $ndq_{sn}$
PROCEDURE:
1: for $i$ = 1 to $n$ do
2:     $p_i \leftarrow$ count_decision_parameters $(DQ_i)$
3:     $l \leftarrow$ rds($H[p_i]$)    ▷ rds($H[p_i]$): function to retrieve decisions from the hash table
4:     $A = \phi$    ▷ $A$: null string of maximum length $n$
5:     for Each Decision $D_j \in l$ do
6:         for Each Decision $D_k \in l$ do
7:             if $\mathbb{M}[D_j][D_k] == p_i$ then
8:                 $A.append(D_k)$
9:             end if
10:        end for
11:    end for
12:    for each Decision $D_j \in A$ do
13:        if $D_j = DQ_i$ then
14:            return decision of $DQ_i$ to user
15:            if $DQ_i.type = EDQ$ then
16:                $EDQ.dequeue(DQ_i)$
17:                $ndq_{sn} \leftarrow ndq_{sn} - 1$
18:            else
19:                $NEDQ.dequeue(DQ_i)$
20:                $ndq_{dv} \leftarrow ndq_{dv} - 1$
21:            end if
22:        else
23:            if $DQ_i.type = EDQ$ then
24:                execute $DQ_i$ in sensor network
25:            end if
26:        end if
27:    end for
28: end for
```

Proposition 2. *The running time complexity of DQ-Class is $O(n)$, where n represents the total number of decision queries requested by the end-users at any time instant t.*

Proof: The algorithm, DQ-Class, classifies the queries requested by the end-users into two queues – emergency (EDQ) and non-emergency ($NEDQ$) queue, which is performed by the enqueue operation. The loop used to classify the decision queries iterates n number of times. During each iteration, the loop compares the DQ type, enqueue it based on the type, and increment the queue length in the $O(1)$. Therefore, the total running time complexity of DQ-Class is $O(n)$. ■

Algorithm 2 maps the DQs requested by the end-users with the stored decisions in the decision database. During this stage, the parameters of a DQ are checked with the parameters of already generated decisions stored in the database, to avoid repetitive execution of the same query. Further, we maintain a strict ordering among the decision parameters to nullify the reordering of the decision parameters, during the computation of similarity among the DQs.

Proposition 3. *The running time complexity of DQ-Map is $O(nm^2)$, where n represents the total number of decision queries requested by the end-users and m denotes the number of decisions stored in the database, at any time instant t.*

Proof: The proposed scheme, DQ-Map, matches the decision query (DQ) request from the stored decisions in the second level. The outer loop of this algorithm iterates n number of times for n DQs. In Step -3 of Algo 2,we use the rds function to retrieve the decisions from the hash table on the basis of parameter count. The running time of this operation is of the order of $O(1)$. The other operations enqueue, dequeue, and append represent the insert, delete, and arrange the elements at the end of the queue. These operations have time complexity of $O(1)$. We consider that there are m number of stored decisions present in the database. Therefore, the maximum length of the list, l, which stores the decisions retrieved from the hash table, is m. The time complexity is $O(m)$. Algorithm 2 is used to filter the retrieved list. In the worst case, the lines $5 - 11$ of Algorithm 2 has running time complexity of $O(m^2)$. The maximum length of the null string, A, is m, which stores the filtered list. The complexities of serving the DQs on the basis of similarity (line $12 - 27$) is $O(m)$. Therefore, the total running time complexity of Algorithm 2 is represented as: $max(nm^2, nm) \implies O(nm^2)$. ■

The non-emergency decision queries which are not served from the decision database are forwarded to the final stage of mapping to find out the similarity among them. Algorithm 3 represents the generation of new queries on the basis of similarities among the decision queries.

Proposition 4. *The running time complexity of DQ-Sim is $O(cn^2)$, where c denotes the total number of decision parameters available with Safe-aaS and n represents the total number of decision queries requested by the end-users, at any time instant t.*

Proof: The algorithm 3 reforms the NEDQs on the basis of similarity among them. The maximum size of each decision query is c. So, the worst case time complexity of two decision query intersection and difference is $O(c)$. The time complexity based on the maximum size of the non-emergency queue is $O(n)$. Therefore, the run-time complexity of DQ-Sim is

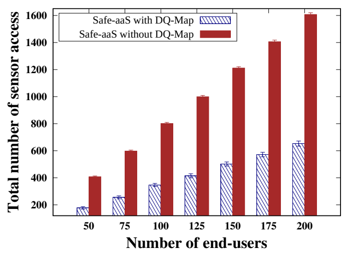

Fig. 2: Total sensor access

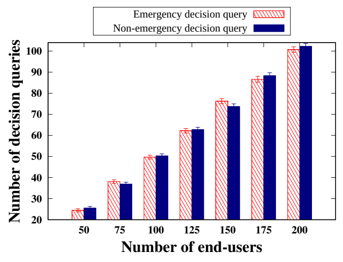

Fig. 3: Number of decision queries

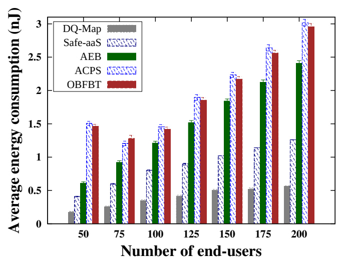

Fig. 4: Energy consumption

$O(cn^2)$, in the worst case. ■

Complexity analysis: Algorithm 1 classifies the DQs as – EDQ and NEDQ, and it takes $O(n)$ time. The Algorithm 2 maps the DQs with the stored decisions in the database and runs in $O(nm^2)$. Further, in case of mismatch with the stored decisions, Algorithm 3 finds the similarity among the NEDQs and takes $O(cn^2)$. Therefore, the total running time complexity of the proposed scheme is $O(n) + O(nm^2) + O(cn^2) \approx O(n(m^2 + n))$.

Algorithm 3 DQ-Sim

INPUT: NEDQ
OUTPUT: NEDQ with similarity among the queries
PROCEDURE:

```
1:  for each $DQ_j$ and $DQ_k$ in NEDQ do
2:      if $DQ_j == DQ_k$ then
3:          NEDQ.$dequeue(DQ_k)$
4:      else
5:          $DQ_{jk} \leftarrow DQ_j \cap DQ_k$
6:          $DQ_{j1} \leftarrow DQ_j \setminus DQ_{jk}$
7:          $DQ_{k1} \leftarrow DQ_k \setminus DQ_{jk}$
8:          NEDQ.$dequeue(DQ_j, DQ_k)$
9:          NEDQ.$enqueue(DQ_{j1}, DQ/_{k1}, DQ_{jk})$
10:     end if
11: end for
12: Forward NEDQs to sensor network for execution
```

TABLE I: Simulation Parameters

Parameter	Value
Simulation area	$10\ km \times 10\ km$
Number of decision parameters	15
Number of sensor nodes	500
Number of decisions in the database	5 – 15
Number of end-users	50–200
Initial energy of sensor nodes	2 nJ
Deployment type	random

V. Performance Evaluation

A. Simulation Design

In this section, we evaluate the performance of our proposed scheme, DQ-Map, we consider the presence of 500 randomly deployed sensor nodes over a simulation area of $10 \times 10 km^2$. We consider the presence of five types of static and mobile

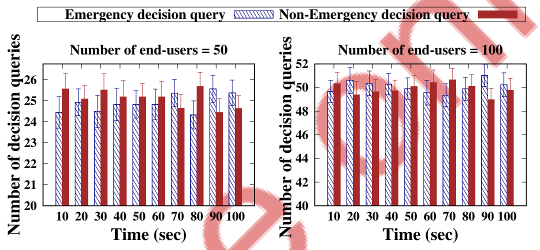

Fig. 5: Variation of EDQ and NEDQ

sensor nodes in the simulation environment. Static sensor nodes are deployed at a particular geographical location, while the mobile sensor nodes are externally placed into the vehicles. We apply Gauss Markov mobility model to design the speed v_n and direction d_n of the mobile sensor nodes at the n^{th} time instant. Mathematically,

$$v_n = \alpha v_{n-1} + (1-\alpha)\bar{v} + \sqrt{(1-\alpha^2)} \times v_{x_{n-1}} \tag{13a}$$

$$d_n = \alpha d_{n-1} + (1-\alpha)\bar{d} + \sqrt{(1-\alpha^2)} \times d_{x_{n-1}} \tag{13b}$$

where α is the tuning parameter. $\bar{v}$ and $\bar{d}$ denote the mean speed and direction of the mobile sensor node. $v_{x_{n-1}}$ and $d_{x_{n-1}}$ represent the random variable from a Gaussian distribution that assigns randomness to the speed and direction of the sensor node at time instant, $(n-1)$. The different simulation parameters considered are mentioned in the Table I. We execute our experiment upto 100 iterations considering 95% confidence interval.

B. Benchmark

We compared our proposed scheme, DQ-Map, with the existing decision making mechanism [22], traditional Safe-aaS platform [1], driving pattern analysis [23], and vehicle centric safety framework [24]. We termed the traditional Safe-aaS infrastructure as Safe-aaS without DQ-Map and the decision making mechanism for automatic pedestrian braking system as AEB. Further, we termed the vehicle-centric safety framework as ACPS and the on-board feedback-based training analysis as OBFBT. In Safe-aaS [1], the end-users select certain decision parameters and register through the Web portal. Based on these decision parameters, the decision is provided to the end-users. To deliver these decisions to the end-users, we propose a dynamic decision query mapping mechanism. On the other hand, Rosado *et al.* [22] proposed a decision making mechanism for automatic emergency braking systems. The authors validated their proposed model with the test results and it can be applied in real time for its simplicity. However, their proposed decision scheme does not provide a common platform for provisioning customized safety-related decisions to the end-users. Pozueco *et al.* [23] analyzed a road safety monitoring system, which is capable of providing real-time feedback and in-vehicle training. Additionally, the drivers experienced both theoretical and practical sessions, based on a learning system, with on-board feedback techniques during their training. On the other hand, Naufal *et al.* [24] proposed a vehicle-centric safety framework to be applicable for autonomous transportation systems. The framework designed by the authors focuses on minimizing the probability of collision among the vehicles during run-time and risks associated with loss of human life due to accidents. However, these existing schemes do not consider the decision generation through optimal utilization of sensor nodes for provisioning safety-related informations to the end-users. Figs. 4 and 8 illustrate the variations in the energy consumption and decision generation time with the increase in number of end-users in the proposed scheme, DQ-Map, traditional Safe-aaS, AEB, ACPS, and OBFBT. We observe that the energy consumption is improved and decision generation time is minimized using DQ-Map.

C. Results

The various performance evaluation parameters are discussed as follows:

Total sensor access: Fig. 2 illustrates the variation of total number of sensor access in the Safe-aaS infrastructure with the proposed scheme, DQ-Map, and without the scheme. We increase the number of end-users along the x-axis from 50 upto 200, with an interval of 25. Interestingly, we observe that the total number of sensor access, $\mathbb{TA}_t$, increases with the increase in the number of end-users. However, we observe that the rate of increase in $\mathbb{TA}_t$ is lower in Safe-aaS with DQ-Map. The possible reason behind such trend is the similarity of the DQs with the stored decisions and similarity among the NEDQs, in case of DQ-Map.

Number of decision queries: Figs. 3 and 5 depict the variations in the number of decision queries (DQs) with the increase in the number of end-users and time. In Fig. 3, we vary the number of end-users along the x-axis. We observe an increasing trend in both the emergency decision query (EDQ) and non-emergency decision query (NEDQ). On the other hand, Fig. 5 illustrates the variation in EDQs and NEDQs with time in the presence of 50 and 100 end-users. However, the number of EDQs and NEDQs varies randomly with time.

Energy consumption: Fig. 4 illustrates the average energy consumption in the Safe-aaS infrastructure with DQ-Map and without DQ-Map. We observe that the average energy consumption increases with the increase in the number of end-users. However, the rate of increase in energy consumption is low in the Safe-aaS infrastructure with DQ-Map. On the other hand, the energy consumption of AEB scheme is increased by 32.97%, 70.76%, 86.7%, and 82.89% compared to the traditional Safe-aaS, AEB, ACPS, and OBFBT. The probable reason behind this is the reduction in the number of sensor access, based on the similarity score of decision queries with the stored decisions.

Similarity score: Fig. 6 demonstrates the variations in the similarity score of NEDQs with the increase in the number of end-users. We vary the number of end-users from 50-180 with an interval of 10 along the x-axis. We observe that with the increase in the number of decision parameters in the system, the similarity score among the NEDQ increases. Fig. 7 depicts the similarity score of EDQs and NEDQs with the parameters of the stored decisions. We notice that the similarity score increases with the increase in the number of decision parameters in the system. Further, the similarity score increases with the increase in the number of stored decisions in the database.

Decision generation time: Fig. 8 de Additionally, we compare our proposed scheme with existing schemes such as AEB, ACPS, and OBFBT. We observe that the decision generation time increases with the increase in the number of end-users. However, the rate of increase in decision generation time decreases by 28.56% in case of Safe-aaS with DQ-Map. One of the possible reasons behind this pattern is the reduction in the total number of sensor access, which results in the reduction of overall time consumption in decision generation. On the other hand, we observe that the decision generation time does not fluctuate significantly, in case of AEB. This is because the decision is generated for automatic braking in the presence of pedestrians. add for other existing schemes also

VI. Conclusion

In this paper, we proposed a three-level decision query mapping mechanism, DQ-Map, for Safe-aaS infrastructure. We aim to reduce the number of sensor access to generate customized safety-related decisions, as per the end-user's requests. Further, to provide importance to the emergency queries, we categorize the decisions into emergency (EDQ) and non-emergency (NEDQ) decision query in the first level. In order to avoid the repetitive execution of decision generation, the requested decisions are matched with the existing stored decisions. In case of mismatch with the parameters of stored decisions, EDQs are executed from the underlying sensor network. On the other hand, NEDQs are forwarded to the third level. The similarity among the NEDQs are determined. In case of mismatch, the DQs are executed from the sensor network.

In the future, we plan to design the mapping of decision queries in the presence of malicious and misbehaving nodes. As various actors are involved in the Safe-aaS infrastructure, pricing is an essential part. Thus, we plan to design a dynamic pricing model for the Safe-aaS architecture.

References

[1] C. Roy, A. Roy, S. Misra, and J. Maiti, "Safe-aaS: Decision Virtualization for Effecting Safety-as-a-Service," *IEEE Internet of Things Journal*, vol. 5, no. 3, pp. 1690–1697, June 2018.
[2] R. A. Khalil and N. Saeed, "Network optimization for industrial internet of things (iiot)," *IEEE Sensors Letters*, vol. 4, no. 7, pp. 1–4, 2020.

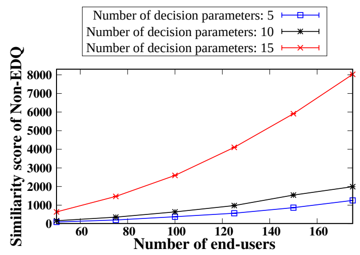

Fig. 6: Similarity score of decision queries

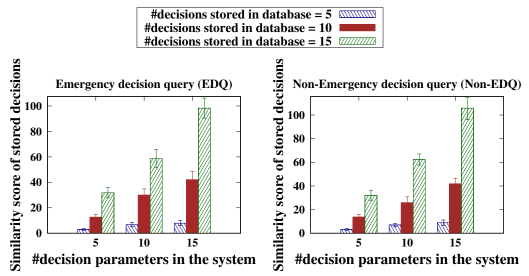

Fig. 7: Similarity score of stored decisions

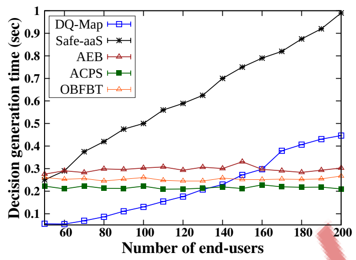

Fig. 8: Decision generation time

[3] M. Wazid, P. Bagga, A. K. Das, S. Shetty, J. J. P. C. Rodrigues, and Y. Park, "AKM-IoV: Authenticated Key Management Protocol in Fog Computing-Based Internet of Vehicles Deployment," *IEEE Internet of Things Journal*, vol. 6, no. 5, pp. 8804–8817, 2019.

[4] G. Al-Kubati, A. Al-Dubai, L. Mackenzie, and D. P. Pezaros, "Stable Infrastructure-Based Routing for Intelligent Transportation Systems," in *IEEE International Conference on Communications (ICC)*, June 2015, pp. 3394–3399.

[5] D. Desiraju, T. Chantem, and K. Heaslip, "Minimizing the Disruption of Traffic Flow of Automated Vehicles During Lane Changes," *IEEE Transactions on Intelligent Transportation Systems*, vol. 16, no. 3, pp. 1249–1258, June 2015.

[6] A. Gregoriades, A. Sutcliffe, G. Papageorgiou, and P. Louvieris, "Human-Centered Safety Analysis of Prospective Road Designs," *IEEE Transactions on Systems, Man, and Cybernetics - Part A: Systems and Humans*, vol. 40, no. 2, pp. 236–250, March 2010.

[7] K. Liang, J. Mrtensson, and K. H. Johansson, "Heavy-Duty Vehicle Platoon Formation for Fuel Efficiency," *IEEE Transactions on Intelligent Transportation Systems*, vol. 17, no. 4, pp. 1051–1061, April 2016.

[8] E. Meissner, T. Chantem, and K. Heaslip, "Optimizing Departures of Automated Vehicles From Highways While Maintaining Mainline Capacity," *IEEE Transactions on Intelligent Transportation Systems*, vol. 17, no. 12, pp. 3498–3511, Dec 2016.

[9] J. Caban, R. Karpiski, and D. Barta, "Road Traffic Accident Injuries Causes and Biomaterial Related Treatment," in *the XI International Science-Technical Conference Automotive Safety*, April 2018, pp. 1–7.

[10] K. T. Chui, K. F. Tsang, H. R. Chi, B. W. K. Ling, and C. K. Wu, "An Accurate ECG-Based Transportation Safety Drowsiness Detection Scheme," *IEEE Transactions on Industrial Informatics*, vol. 12, no. 4, pp. 1438–1452, Aug 2016.

[11] K. C. Dey, A. Mishra, and M. Chowdhury, "Potential of Intelligent Transportation Systems in Mitigating Adverse Weather Impacts on Road Mobility: A Review," *IEEE Transactions on Intelligent Transportation Systems*, vol. 16, no. 3, pp. 1107–1119, June 2015.

[12] S. Fernandez and T. Ito, "Driver classification for intelligent transportation systems using fuzzy logic," in *the* 19^{th} *International Conference on Intelligent Transportation Systems (ITSC)*, Nov 2016, pp. 1212–1216.

[13] D. Geronimo, A. M. Lopez, A. D. Sappa, and T. Graf, "Survey of Pedestrian Detection for Advanced Driver Assistance Systems," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 32, no. 7, pp. 1239–1258, July 2010.

[14] C. Roy, S. Misra, J. Maiti, and M. S. Obaidat, "Dense: Dynamic edge node selection for safety-as-a-service," in *2019 IEEE Global Communications Conference (GLOBECOM)*, 2019, pp. 1–6.

[15] C. Roy, S. Misra, and S. Pal, "Blockchain-enabled safety-as-a-service for industrial iot applications," *IEEE Internet of Things Magazine*, vol. 3, no. 2, pp. 19–23, 2020.

[16] E. Bertolazzi, F. Biral, M. D. Lio, A. Saroldi, and F. Tango, "Supporting Drivers in Keeping Safe Speed and Safe Distance: The SASPENCE Subproject Within the European Framework Programme 6 Integrating Project PReVENT," *IEEE Transactions on Intelligent Transportation Systems*, vol. 11, no. 3, pp. 525–538, Sept 2010.

[17] A. Amditis, E. Bertolazzi, M. Bimpas, F. Biral, P. Bosetti, M. D. Lio, L. Danielsson, A. Gallione, H. Lind, A. Saroldi, and A. Sjogren, "A Holistic Approach to the Integration of Safety Applications: The INSAFES Subproject Within the European Framework Programme 6 Integrating Project PReVENT," *IEEE Transactions on Intelligent Transportation Systems*, vol. 11, no. 3, pp. 554–566, Sept 2010.

[18] C. Roy, S. Misra, J. Maiti, and U. Chakravarty, "Safe-Serv: Energy-Efficient Decision Delivery for Provisioning Safety-as-a-Service," *IEEE Transactions on Services Computing*, pp. 1–1, 2020.

[19] C. Roy and S. Misra, "Safe-Passe: Dynamic Handoff Scheme for Provisioning Safety-as-a-Service in 5G-Enabled Intelligent Transportation System (accepted)," *IEEE Transactions on Intelligent Transportation Systems*, 2021.

[20] M. Fazeen, B. Gozick, R. Dantu, M. Bhukhiya, and M. C. Gonzlez, "Safe Driving Using Mobile Phones," *IEEE Transactions on Intelligent Transportation Systems*, vol. 13, no. 3, pp. 1462–1468, Sep. 2012.

[21] M. S. Bazaraa, *Nonlinear Programming: Theory and Algorithms*, 3^{rd} ed. Wiley Publishing, 2013.

[22] A. Lpez Rosado, S. Chien, L. Li, Q. Yi, Y. Chen, and R. Sherony, "Certainty and Critical Speed for Decision Making in Tests of Pedestrian Automatic Emergency Braking Systems," *IEEE Transactions on Intelligent Transportation Systems*, vol. 18, no. 6, pp. 1358–1370, 2017.

[23] L. Pozueco, N. Gupta, X. G. Paeda, R. Garca, A. G. Tuero, D. Melendi, A. Rionda, and V. Corcoba, "Analysis of Driving Patterns and On-Board Feedback-Based Training for Proactive Road Safety Monitoring," *IEEE Transactions on Human-Machine Systems*, vol. 50, no. 6, pp. 529–537, 2020.

[24] J. K. Naufal, J. B. Camargo, L. F. Vismari, J. R. de Almeida, C. Molina, R. I. R. Gonzlez, R. Inam, and E. Fersman, "A2CPS: A Vehicle-Centric Safety Conceptual Framework for Autonomous Transport Systems," *IEEE Transactions on Intelligent Transportation Systems*, vol. 19, no. 6, pp. 1925–1939, 2018.