Data federation through on-demand queries in intelligent transport systems

O K Golovnin
Department of Information Systems and Technologies, Samara University, 34, Moskovskoe shosse, Samara, 443086, Russia

golovnin@ssau.ru

Abstract. The paper proposes a federated approach to data integration in intelligent transport systems when data are requested from various sources using the on-demand query technique rather than traditional approach when data are accumulated in a single storage. The proposed approach is designed to reduce the number of data manipulations, thereby increasing the processing speed and reducing the necessary computing resources. The study was carried out on synthetically generated data: 10 data sources are used; each of them consists of 650000 records. Data federation reduces the memory used by an average of 53%, while the time of virtual federal database creation is on average 71% less than the traditional integration approach implementation. The approach to data federation is designed for implementation in intelligent transport systems of a new generation, where it is required to process large amounts of data coming from various heterogeneous sources.

1. Introduction
With the growth of the amount and complexity of data using by modern intelligent transport systems (ITS), the need for developing approaches for collecting, processing, analyzing, and storing data increases, that is, effective data manipulation in ITS comes to the fore [1, 2]. So, ITS rely on detailed data containing information not only about the characteristics of traffic flows but also about transport infrastructure facilities, technical means of traffic control, various road, weather, and social events leading to a change in traffic [3, 4]. Thus, the sources of data for ITS are many urban information systems, technical and personal mobile devices, which often duplicate data on the same characteristics used by ITS. In addition, due to the specifics of the ITS operation, data processing must be online, which can be difficult with a large amount of data.

In this work, the author proposes a federated approach to data integration in intelligent transport systems, in which data is not accumulated in a single storage, but is requested from sources, if necessary, using the query technique. The proposed approach is designed to reduce the number of data manipulations, thereby increasing the processing speed and reducing the necessary computing resources that are used in data processing in ITS.

2. State of the art
The problem of data integration in ITS has been considered by many researchers. In [5], the authors proposed a two-tier architecture for collecting, storing, processing, and integrating information in real-time, however, the implementation of the proposed solution requires significant changing of existing
ITS subsystems, which is associated with colossal time, organizational, financial and technical difficulties. In [6], the features to reduce the amount of transmitted data in the conditions of ITS are considered, including the proposed method based on indexing database tables, which, however, cannot be applied in the conditions of modern cities. For example, in [7] it is shown that the basis of ITS functioning is now formed by “non-traditional” data generated by systems of the “Smart City” class, mobile applications, data from social networks and, in general, any devices in the concept of the Internet of Things.

Another feature of the ITS is the need for a system architecture that would provide scalability, that is, the connection of new data sources (subsystems) for the ITS [8]. First of all, the rapid implementation of new data sources is facilitated by the development of technologies for data mining and machine learning [9]. The development of intelligent technologies makes it possible to overcome direct and indirect problems of obtaining data on traffic and events affecting it, thus, in the short term, these are one of the most promising data sources, in which data are presented in a structure that is not traditional for ITS. Taking this into account, the data integration system for ITS should support complex data structures coming from video analytics systems [10], audio analytics [11], crowdsourcing systems [12], and others [13-15].

An effective strategy for data integration in such conditions is federation, in which the raw data remains in the systems where it originated, that is, the data is not copied to the ITS database. Classical approaches to the federation of databases cannot be applied in modern ITS since they operate with various heterogeneous data sources [16]. A promising approach is seen in which virtual storage is created that supports SQL-like queries and presents data from several data sources [17]. In this case, the presentation of data can be implemented in the form of patterns [18, 19], which provide both flexibilities in the structure of the data record and in its attribute composition.

In this paper, the author proposes the data federation approach based on structured queries on demand. The results are proposed to be used as part of modern ITS, where traffic control is based on data generated by a multiple of interacting subsystems.

3. On-demand queries for data federation
Let's represent one data record $R_j$ with the following tuple:

$$ R_j = (r_1, r_2, ..., r_n), \forall r_i \in D, $$

(1)

where:
- $D$ – the domain of the data type;
- $R_j$ – an attribute with $i$ number;
- $n$ – total count of attributes;
- $j$ – current record number.

The domain of data type $D$ for an attribute is represented by one of the following sets:

$$ D = \emptyset \cup ((\mathbb{Z} \subset \mathbb{R}) \cup (C \subset L)), $$

(2)

where:
- $\emptyset$ – an empty set;
- $\mathbb{Z}$ – a set of integers;
- $\mathbb{R}$ – a set of real numbers;
- $C$ – a set of characters: $C = \{"A","B","C",...,"z","\ldots\}\};$
- $L$ – a set of strings: $L = \{l_1, l_2, ..., l_n\}, \forall l_i \in C$.

The dataset of one source is denoted as:

$$ R^k = (R_1, R_2, ..., R_m), $$

where:
- $R_j$ – a record in the source;
$k$ – source number: $k = 1, t$, where $t$ is the total number of sources;
$m$ – the number of data records in the source.
Thus, the statement of the integration problem is as follows:

$$\hat{R} = \bigcup_{k=1}^{t} R^k.$$  

When solving the integration problem in a classical way, the following operation is performed:

$$\hat{R} = \bigcup_{k=1}^{t} \bigcup_{j=1}^{m} \bigcup_{i=1}^{n} (g(r_i) \in R^k_j),$$  \hspace{1cm} (3)

where $g(r_i)$ – is a preprocessing operation performed on each attribute. In the simplest case, when data preparation operations are not required: $g(r_i) = r_i$. In the case when it is required to process complex records:

$$g(r_i) = \begin{cases} 
\eta_i, & \text{if } r_i \in D \setminus L; \\
\bigcup_{l_j \in r_i} g(l_j), & \text{if } \eta_i \in L; \\
\bigcup_{r_a \in r_i} g(r_a), & \text{if } r_i \notin D.
\end{cases}$$

When implementing the principle of federation, data unification is represented as follows:

$$\hat{R} = \bigcup_{k=1}^{t} \bigcup_{j=1}^{m} f(k, j).$$  \hspace{1cm} (4)

That is, this way is replaced the direct processing with reflection through on-demand queries:

$$f: k, j \rightarrow \bigcup_{i=1}^{n} (g(r_i) \in R^k_j),$$

$$f^{-1}((r_1, r_2, ..., r_n)) = \{(k, j) \in t \times m | f(k, j) = (r_1, r_2, ..., r_n)\}.$$ 

And define data retrieval operation as follows:

$$R^* = \{r_i \in \hat{R} | P(r_i)\},$$  \hspace{1cm} (5)

where $P(r_i)$ is a projection.

4. Results
Data federation is implemented on the .NET Framework 4.7.2. The tests are carried out on a workstation running Windows 10. The workstation is equipped with 32 GB of DDR4-2933 RAM, an Intel Core i9-10900 processor (up to 5.2 GHz, 10 cores, 20 threads), an Nvidia GeForce RTX 2070 Super video card, and 1 TB SSD. PostgreSQL 13.0 with standard configuration settings was used as a relational data storage. During data integration operations, the used memory and the operation time are measured.

The study was carried out on the following synthetic test data. 10 data sources are used. The number of data records ranges from 50000 to 650000. The length of data record $R_j$ (1) is random but does not exceed 10 attributes. The domain $D$ (2) of an attribute is evenly distributed among all its types. The length of the strings is random but does not exceed 255 characters. The characters in the standard UTF-8 set are used.

As an alternative to federation (4), classical integration (3) is used, hereinafter there is a comparison with it.

The results of measuring memory usage are shown in figure 1.
Figure 1. Memory usage depending on records count (simple records).

Figure 1 shows that traditional integration consumes more memory than the federation. As the number of records increases, the difference in used memory becomes more and more significant. On average, federalization shows 53% of memory savings.

The results of measuring the execution time when building a database are shown in figure 2.

Figure 2. Operation time depending on records count (simple records).

Figure 2 shows that traditional integration takes more time to build a database than federation. On average, federation shows 71% operation time savings.
The second step of study is done with the complexity of the data records (1): the data record can include links to other records, i.e. domain set (2) extension is proposed:

\[ \exists r_a \not\in D, \quad r_a = \langle r_1, r_2, ..., r_n \rangle. \]

Domains are distributed evenly. The number of data records ranges from 50000 to 450000.

In this case, the advantage of federalization becomes more and more noticeable, especially in the used memory (figure 3). On average, federation requires 72% less memory.

![Figure 3. Memory usage depending on records count (complex records).](image)

The nature of the operation time changing has been saved (figure 4): federation shows an average advantage of 75%.

![Figure 4. Operation time depending on records count (complex records).](image)

The study on data retrieval shows that the execution time of a simple query (5) is equivalent: simple queries to retrieve data against a virtual database (federation) are faster than similar queries against a traditional database (integration), but the execution time differs on average only by 3%. 

5
5. Conclusions
The approach to data integration presented in the work can be used in intelligent transport systems and situational centers of a new generation, where it is required to process large amounts of data coming from various sources and partially duplicating each other. The use of data federation reduces the memory used by an average of 53% during the operation of the ITS, while the time to create a virtual federal database is on average 71% less than when using the traditional approach. At the same time, simple queries to retrieve data against a virtual database are faster than similar queries against a traditional database, because in virtual database queries are not performed on relational tables with a large number of records.

Further work on this topic will be aimed at developing ways to ensure the reliability and relevance of data since the same characteristic of the traffic and events can be represented by data from sources that are different in their adequacy. Another issue requiring further development is securing federalized data, since different sources may have different access rights, they will also require integration.

References
[1] Telang S, Chel A, Nemade A and Kaushik G 2020 Intelligent Transport System for a Smart City Security and Privacy Applications for Smart City Development (Cham: Springer) pp 171–87
[2] Kvet M and Matiaško K 2018 Analysis of temporal data management in the intelligent transport system World Symp. on Digital Intelligence for Systems and Machines pp 151–8
[3] Surmin O, Sitnikov P, Suprun A, Ivaschenko A, Stolbova A and Golovnin O 2019 Urban Public Transport Digital Planning based on an Intelligent Transportation System Conf. of Open Innovations Association FRUCT (Helsinki: IEEE) pp 292–8
[4] Golovnin O and Mikheeva T 2017 Detailed Models and Network-Centric Technologies of Transport Process Management Int. Conf. on Models and Technologies for Intelligent Transportation Systems (Napoli: IEEE) pp 768–73
[5] Khan N A, Brujic-Okretic V and Khaddaj S 2016 Intelligent integration framework for smart transport system Int. Conf. on Intelligent Environments pp 76–9
[6] Kvet M, Matiaško K and Kvet M 2016 Temporal index dispatcher layer service for intelligent transport system ELEKTRO pp 417–22
[7] Karouani Y and Elhoussaine Z 2017 Toward an intelligent traffic management based on Big Data for Smart City Proc. of the Mediterranean Symposium on Smart City Applications (Cham: Springer) pp 502–14
[8] Luckow A and Kennedy K 2017 Data infrastructure for intelligent transportation systems Data Analytics for Intelligent Transportation Systems pp 113–29
[9] Alsrehin N O, Klaib A F and Magableh A 2019 Intelligent transportation and control systems using data mining and machine learning techniques: A comprehensive study IEEE Access 7 49830–57
[10] Kavitha N and Chandrappa D N 2017 Comparative review on video based vehicular traffic data collection for intelligent transport system Int. Conf. on Recent Advances in Electronics and Communication Technology pp 260–4
[11] George J, Cyril A, Koshy B I and Mary L 2013 Exploring sound signature for vehicle detection and classification using ANN Int. J. on Soft Computing 4(2) 29
[12] Kumar H, Singh M K and Gupta M P 2018 Smart mobility: Crowdsourcing solutions for smart transport system in smart cities context Proc. of the 11th Int. Conf. on Theory and Practice of Electronic Governance pp 482–8
[13] Ashokkumar K, Sam B and Arshadprabhu R 2015 Cloud based intelligent transport system Procedia Computer Science 50 58–63
[14] Qureshi K N and Abdullah A H 2013 A survey on intelligent transportation systems Middle-East J. of Scientific Research 15 629–42
[15] Rathi R 2016 Intelligent transport system for the development of smart cities Int. J. of Transportation Engineering and Traffic System 2(2) 20–9
[16] Haas L M, Lin E T and Roth M A 2002 Data integration through database federation IBM Systems J. 41(4) 578–96
[17] Vu X S, Ait-Mlouk A, Elmothe E and Jiang L 2019 Graph-based interactive data federation system for heterogeneous data retrieval and analytics The World Wide Web Conf. pp 3595–9
[18] Mikheeva T, Mikheev S and Golovnin O 2019 Pattern Model Stratification to Decision-Making Support of Transport Infrastructure Management Proc. of the 5th Int. Conf. on Vehicle Technology and Intelligent Transport Systems pp 498–504
[19] Liu S, Huang X, Fu H and Yang G 2013 Understanding data characteristics and access patterns in a cloud storage system Int. Symp. on Cluster, Cloud, and Grid Computing pp 327–34