Jurnal Teknologi

Full paper

Big Data Processing and Mining for Next Generation Intelligent Transportation Systems

Jelena Fiosina[a*], Maxims Fiosins[a], Jörg P. Müller[a]

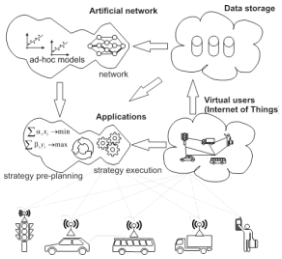
Clausthal University of Technology, Institute of Informatics, Julius-Albert Str. 4, D-38678, Clausthal-Zellerfeld, Germany

*Corresponding author: Jelena.Fiosina@gmail.com

Article history

Received :11 September 2012
Received in revised form :
21 February 2013
Accepted :15 April 2013

Graphical abstract



Abstract

The deployment of future Internet and communication technologies (ICT) provide intelligent transportation systems (ITS) with huge volumes of real-time data (Big Data) that need to be managed, communicated, interpreted, aggregated and analysed. These technologies considerably enhance the effectiveness and user friendliness of ITS, providing considerable economic and social impact. Real-world application scenarios are needed to derive requirements for software architecture and novel features of ITS in the context of the Internet of Things (IoT) and cloud technologies. In this study, we contend that future service- and cloud-based ITS can largely benefit from sophisticated data processing capabilities. Therefore, new Big Data processing and mining (BDPM) as well as optimization techniques need to be developed and applied to support decision-making capabilities. This study presents real-world scenarios of ITS applications, and demonstrates the need for next-generation Big Data analysis and optimization strategies. Decentralised cooperative BDPM methods are reviewed and their effectiveness is evaluated using real-world data models of the city of Hannover, Germany. We point out and discuss future work directions and opportunities in the area of the development of BDPM methods in ITS.

Keywords: Cloud computing architecture; ambient intelligence; big data processing and mining; multi-agent systems; distributed decision-making

1.0 INTRODUCTION

Increasing traffic and frequent congestion on today's roads require innovative solutions for infrastructure and traffic management. As the components of traffic systems become more autonomous and smarter (e.g. new communication capabilities of vehicles and infrastructure), there is an increasing need for cooperation among intelligent transportation systems (ITS) for transportation management and environmental monitoring in order to improve traffic management strategies. Further, there is growing interest and increasing volume of investments to smarter transportation management systems. In these new-generation business management systems, the management of transportation networks is closely integrated with the business strategies and operational models of transport companies and individual customers, providing a considerable impact for companies in terms of business planning, service quality and adaption to customer needs as well as for individual users in terms of time and money saving, adaptive travel planning and support of mutually beneficial social behaviour.

All participants of ITS act as data generators and sources leading to a huge amount of available data with a short update rates. This growth in data production is being driven by: individuals and their increased use of media (social networks); novel types of sensors and communication capabilities in vehicle and the traffic infrastructure; application of modern information and communication technologies (ICT) (*Cloud computing, Internet of Things (IoT), etc.*) with the proliferation of internet connected devices and systems. Data sets grow in size because they are being gathered increasingly by ubiquitous information-sensing mobile devices, aerial sensory technologies (remote sensing), software logs, cameras, microphones, radio-frequency identification readers, and wireless sensor networks [1]. There has also been acceleration in the proportion of machine-generated and unstructured data (photos, videos, social media feeds, etc.). Thus, there is an emerging Big Data problem in ITS. Big Data usually includes data sets with sizes beyond the ability of commonly used software tools to capture, curate, manage, and process the data within a tolerable elapsed time [2]. Big Data supplies more detailed information about the customers and their behaviour, but should be properly analysed in a decentralized (multi-agent) fashion while avoiding transmission of big information volumes. Therefore cloud and grid computing infrastructures are well-suited for storing, management and processing of Big Data.

By nature, Big Data is physically and logically decentralized, but virtually centralized. All information sources/storages are interconnected, and any piece of information can in principle be accessed by any component of the system. Big Data volumes are (very often) created and managed in a

decentralized fashion on the physical layer. This raises the information costs that should be taken into account while accessing the information.

To find an effective balance between decentralized information processing/decisions and costs of data transfer/decision coordination, Big Data processing and mining (BDPM) as well as corresponding decision-making methods are required. This creates a need to employ innovative BDPM and to develop corresponding decision-making algorithms to support ITS applications in finding, collecting, aggregating, processing, and analysing information that is necessary for optimal decision-making and effective user behaviour strategies.

Modern ICT technologies are used to solve the problems of BDPM employment more effectively. Innovative cloud services can be created using the cloud capabilities of future ICT to access smart objects via IoT. This development can enable wide access to necessary information, which is available in-the-cloud. The cloud computing paradigm based on highly scalable, distributed computing resources provides BDPM with big storage and fast computing capabilities. However, implementing a traffic cloud is far from easy. From an end user's point of view, the complexity of data and algorithms is hidden in the cloud. Users (ranging from traffic authorities to car drivers and automated components) expect to work with relatively simple applications and interfaces on the Internet via mobile or embedded devices. These devices are fully connected and can (theoretically) use all the information available from all other users and system elements. This creates great opportunities for coordinated near-optimal management of the infrastructure (e.g. in terms of load balancing).

The contribution of this study is fivefold: First, we analyse related cloud-based architectures and ITS application scenarios. Second, we consider architectures for implementing the corresponding BDPM and decision-making strategies for transportation operations. Third, we discuss the employment of appropriate mathematical methods for three considered scenarios. Fourth, we review decentralised cooperative BDPM methods based computational statistics, and evaluate their effectiveness on real-world transportation data. Fifth, we point out and discuss future work directions and opportunities in the area of the development of BDPM methods in ITS.

The remainder of this paper is organized as follows. Section 2 presents the state of the art in future ICT for ITS architectures. In Section 3, we review BDPM methods and their application to ITS. In Section 4, we introduce a cloud-based data flow architecture for smart transportation networks. In Section 5, we propose and analyse three application scenarios of ITS and consider data analysis and optimization of participants' behaviour strategies in traffic systems. Section 6 reviews cooperative regression and clustering techniques, which establish key tools in a BDMP framework. Section 7 shows experimental results with real-world data for the evaluation of the techniques proposed. Section 8 concludes and discusses future research opportunities.

2.0 STATE OF THE ART

2.1 Motivation and Applications

A strong worldwide interest in opportunities in transportation and mobility has spurred the need for further analysing future ICT capabilities. In Europe, future ICT and IoT research has been a priority direction for the 7th European Framework Programme for Research and Innovation (FP7) and will continue to be so for the upcoming Horizon 2020 Programme (e.g. the objectives 'A reliable, smart and secure IoT for Smart Cities' or 'Integrated personal mobility for smart cities' in FP7 or 'Substantial improvements in the mobility of people and freight' in Horizon 2020). These research questions are motivated and co-funded by private companies and municipalities from the fields of transport, logistics, communication and traffic management (e.g. the FP7 project Instant Mobility [3]). These stakeholders understand the possible enhancements to existing systems that new technologies provide to ITS. Research in this area is still largely at the stage of formulation of scenarios and coordination protocols.

Recently, Big Data has also become a major topic in the field of ICT worldwide. It is evident that Big Data means business opportunities, but also major research challenges. According to McKinsey & Co [2] and Gartner [4], Big Data is 'the next frontier for innovation, competition and productivity'. While US-based companies are widely recognized for their activities in Big Data, very few research organizations are known for their activities and initiatives in this field in Europe. Analytical Big Data services for SME's within Europe are currently non-existing. According to EU White papers [1], there are currently a number of initiatives aimed at adjusting the research landscape to lodge the rapid changes taking place in the processing of data under FP7. There are a number of projects addressing a vast set of topics ranging from content creation and processing, Big Data analytics and real-time processing. In Horizon 2020, Big Data finds its place both in the Industrial Leadership, for example in the activity line 'Content technologies and information management', and in the Societal Challenges, relating to the need for structuring data in all sectors of the economy (health, climate, transport, energy, etc.). Not surprisingly, Big Data is also important to the Excellent Science priority of Horizon 2020, especially on scientific infrastructures and development of innovative high value added services [1].

ITS development is connected with Big Data processing, therefore it is an important application domain for BDPM methods. From the application point of view the following directions are important for ITS [3]. First of all it is personal travel companion, which intends to provide to travellers, surface vehicle drivers and transport operators the benefits of dynamic planning and follow-up of multimodal journeys. The second very important direction considered by many authors is smart city logistics operations, which intend to provide benefits to actors and stakeholders involved in, affected by or dependent on the transportation of goods in urban environments.

2.2 Future ICT for ITS

Future ICT can enhance ITS and provide large-scale infrastructures for high-performance computing that are 'elastic' in nature, by adapting to user and application needs [5].

Modern mobile and communication technologies such as wireless transmission, mobile Internet, mobile sensors as well as mobile devices (e.g. smart phones) serve as a good basis to the ITS. Today, vehicles are equipped with mobile devices with relatively powerful communication and processing capabilities as well as with sensors that provide diverse information about the environment. Constant 3G or 4G mobile Internet connection is nowadays usual and is favourable from price and quality point of view. This means that many traffic participants are already online and interconnected with rapidly growing trend; the problem is how to use this connection in order to provide efficient and reliable on-demand services to traffic participants.

Ambient intelligence (AmI) is a concept of interconnection of sensors and computational resources and using of artificial intelligence methods in order to improve everyday life. AmI is defined as the ability of the environment to sense, adapt, and respond to actions of persons and objects that inhabit its vicinity.

Moreover, the multi-agent system paradigm makes AmI environments act autonomously and socially, featuring collaboration, cooperation, and even competitive abilities. It supports virtual control strategists and management policy makers in decision-making and is modelled using the metaphor of autonomous agents. An architecture of an AmI-enabled ITS is proposed in [63]. Other examples of AmI implementation are intelligent home or intelligent power grids.

Multi-agent systems (MASs) provide a model of networked, cooperative, autonomous systems and provide a suitable metaphor and tools for representing of ITS. MASs consist of multiple autonomous self-interested software entities, called agents. Agents perceive information from the environment, create their own local data models and then make decisions according to their goals and available information. Decisions are then converted to actions, which influence the environment. Agents interact and cooperate on the level of information models (data or model parameter exchange) or on the level of actions (action coordination, group formation) [6]. Multi-agent modelling has been extensively applied for solving transportation problems [7, 8, 9].

Cloud computing systems are oriented towards a high level of interaction with their users, real-time execution of a large number of applications, and dynamic provisioning of on-demand services. In this study, we consider the layered architecture of cloud-based computing systems presented in [10]. It supports a class of specialized distributed systems that is characterized by a high level of scalability, service encapsulation, dynamic configuration, and delivery on demand. Beside that transport infrastructure can be considered as a service, which studies possibilities how to use cloud data storage, cloud computing virtualization or services-in-the-cloud. The complexity of cloud-based systems is hidden from end users.

Agent-based cloud computing is a paradigm that identifies several common problems and provides several benefits by the synergy between MASs and cloud computing. Cloud computing is mainly focused on the efficient use of computing infrastructure through reduced cost, service delivery, data storage, scalable virtualization techniques, and energy efficiency. In contrast, MAS are focused on intelligent aspects of agent interaction and their use in developing complex applications. In particular, cloud computing can offer a very powerful, reliable, predictable and scalable computing infrastructure for the execution of MASs by implementing complex, agent-based applications for modelling and simulation. Also, software agents can be used as basic components for implementing intelligence in clouds, making them more adaptive, flexible, and autonomic in resource management, service provisioning and large-scale application executions [5].

IoT provides a new approach for virtual representation of real-world entities in virtual environment (e.g. vehicles in the cloud-based ITS). It is a very important aspect for constructing of cloud-based systems [11]. IoT semantically means a world-wide network of interconnected objects (radio frequency identification, infrared sensor, global positioning system, laser scanner, etc.) uniquely addressable that ensure the exchange and sharing of information in ITS. The basic idea of this concept is the pervasive presence around us of a variety of things or objects (radio frequency identification tags, sensors, actuators, mobile vehicles, etc.), which, through unique addressing schemes, are able to interact with each other and cooperate with their neighbours to reach common goals. IoT provides for ITS two main things: 1) its data acquisition function provides more comprehensive traffic data; 2) provides a good channel for traffic data transmission. Therefore, ITS based on IoT has broad prospects of development and expansion space [11].

2.3 ITS Architectures

A 'V-cloud' architecture was proposed in [12], which considers cloud computing in ITS from device, communication and service level the points of view. It facilitates the interaction between vehicle drivers and outside car world, taking into account vehicle-to-vehicle (V2V) and vehicle-to-infrastructure (V2I) interactions to share and utilize external resources in a more effective way (Figure 1).

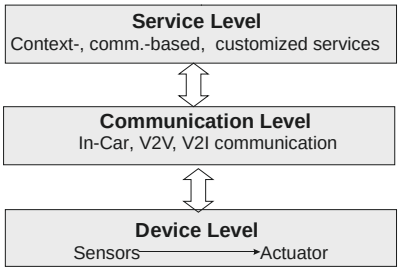


Figure 1 V-cloud' ITS architecture

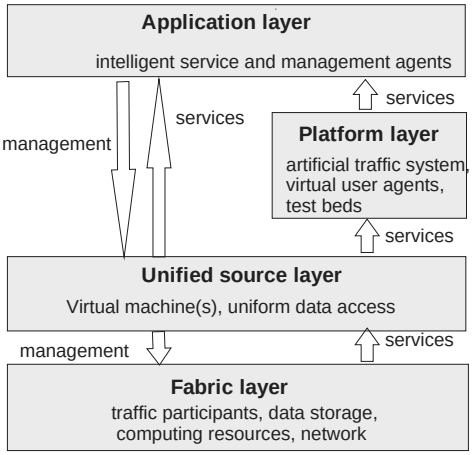


Figure 2 Cloud-based ITS architecture

A layered ITS architecture based on cloud computing paradigm is considered in [13] (Figure 2). The layers of this architecture are listed below.

The **fabric** layer includes all computing, storage, data, and network resources available in the cloud. The resources are accessible through the resource services, are used for cloud computations, management, and as test beds.

The **unified** source layer provides infrastructure-as-a-service by defining unified access to the raw computational resources of the fabric layer using a virtual machine.

The **platform** layer provides platform-as-a-service, including a collection of specialized tools, middleware, and services on the top of unified resources to create a deployment platform (e.g. scheduling create service and artificial test beds).

The **application** layer contains all applications that are run in the cloud. Application execution in the cloud is distributed: applications can be partly executed on the client, partly in the cloud.

The application of cloud-based architectures for ITS is demonstrated in [13]. In order to provide an acceptable level of service, a cloud-based ITS consists of two main components: an *application component*, which provides dynamic services and

runs all the cloud applications; and a *digital (simulated) traffic network* component, which performs constant information collection and processing in order to provide in-time data. A cloud-based ITS adapts its decisions by using available information and by interacting with human as well as automated traffic participants.

The logical architecture of ICT-enabled ITS was considered in [14], where data and computations can use traffic clouds. This research centres on the *ACP (artificial, computational, parallel)* approach. It involves modelling by artificial systems, analysis by computational experiments, and operation through parallel execution for control and management of complex systems with social and behavioural dimensions. ACP-based framework is presented, which is a generalization of the feedback control mechanism in the control theory. The actual system and its artificial counterparts can be connected in various modes (learning and training, experimentation and evaluation, control and management) for different purposes (Figure 3). Artificial transportation system is developed to create a dynamic or 'living' ontology to present and organize transportation knowledge, such as methods, algorithms, regulations, and case studies in a way that's effective for search and ready for computing and implementation. By comparing and analysing of real and simulated behaviours, systems' future actions can be learned and predicted; control and management strategies for their operations can be accordingly planed and modified. The most interesting mode for our study is the Learning and Training mode. In this mode, the artificial systems serve mainly as a data centre for learning operational procedures and for training operators and administrators (Figure 4).

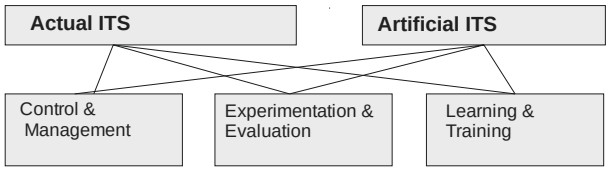


Figure 3 ACP-based ITS architecture

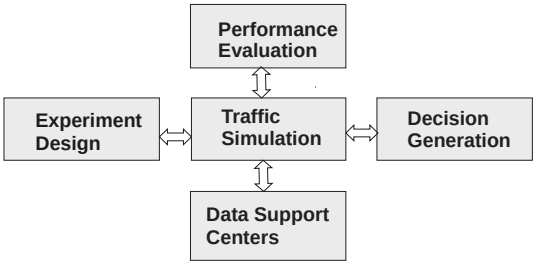


Figure 4 Architecture of the Learning and Training block

A cloud-based traffic network, which employs ambient intelligence (AmI) [63] or IoT components, is described in [10].

In our previous works, we investigated MAS architectures and decision-making methods [15, 16] as well as data processing stages for a typical cloud-based ITS [17], distributed data processing, mining including methods of computational statistics [18, 19] for existing transportation problems. In the following sections we give an overview of core BDPM methods and technologies used in ITS as well as consider sample ITS architecture [17] with corresponding data flows and their processing stages.

3.0 BDPM for ITS

3.1 BDPM Methods

Classical methods of data processing and mining are centralized: this means that in order to apply them, data must be available here and now. In opposite, Big Data is constantly updated and collected in physically distributed storages, and data centralisation is not possible.

Using the centralized approach the system cannot adapt quickly to situations in real time, and it is very difficult or simply impossible to transmit Big Data over the network and to store, manage and process large data sets in one location [20, 16]. In addition some nodes of the distributed system prefer to relay mostly of their own experience in the prediction process. Therefore there is an inherent need to develop effective BDPM algorithms using decentralised architecture that takes into account space and time distribution of data.

Analysing and processing Big Data is now feasible both from a technical and cost perspective. Many Big Data frameworks are built around an understanding of business mechanics, analysis of the business strategy, identifying value and correlation in unstructured and structured data, data mining, predictive analysis and cost effective data [1]. BDPM methods help to store Big Data in a compact way (clustering) by dimension reduction finding rules in data behaviour by predictive modelling, filtering and detecting outliners by change point analysis. The focal BDPM methods are briefly characterized below:

Regression analysis is the most widely used statistical tool for discovering the relationships among variables that is often used for forecasting or prediction. This group of statistical techniques determines, how the value of the dependent variable changes when the values of one or more independent variables are modified [21].

Time series analysis is a set of statistical techniques to model and explain time-dependent series of data points. Time series forecasting uses a model to generate predictions (forecasts) for future events based on known past events. Time series data has a natural temporal ordering - this differs from typical data mining/machine learning applications, where each data point is an independent realisation of the concept to be learned, and the ordering of data points within a data set does not matter [22].

Cluster analysis is a set of statistical methods for classifying objects that splits a diverse group into smaller groups of similar objects, whose characteristics of similarity are not known in advance. This is a type of unsupervised learning because training data are not used [2, 23].

Classification is a set of techniques to identify the categories in which new data points belong, based on a training set containing data points that have already been categorized. These techniques are often considered as supervised learning because of the existence of a training set [24].

Change-point analysis is a powerful, robust and flexible tool, which well characterises changes, determining whether a change has taken place. It is capable of detecting subtle changes; it characterizes the changes detected by providing confidence levels and confidence intervals. It suits also well for outliner detection. Change-point analysis suites well for processing historical data, especially when dealing with Big Data [25].

3.2 MAS for BDPM

Usually MAS represents a complex system, which consists of a large number of autonomous interacting components. Such systems are usually characterized by Big Data, represented by huge volumes of distributed data from various sources. One of

the key challenges in MASs is the capability of agents to process and mine such distributed data in order to provide sufficient information for optimal decisions.

BDPM provides algorithmic solutions for data analysis in a distributed manner to detect hidden patterns in data and extract the knowledge necessary for decentralised decision-making [26, 27]. BDPM methods improve agent intelligence and MAS performance [28], involving pro-active and autonomous agents that perceive their environment, dynamically reason out actions on the basis of the environment, and interact with each other. Furthermore, the coupling of MAS with BDPM may be described in terms of ubiquitous intelligence [29], with the aim of fully embedding information processing into everyday life. Klusch at al. [30] conclude that autonomous data mining agents, as a special type of information agents, may perform various kinds of mining operations on behalf of their user(s) or in collaboration with other agents. Systems of cooperative information agents for BDPM in distributed, heterogeneous, or homogeneous, and Big Data environments appear to be quite a natural progression for the current systems to be realised in the near future.

In many complex domains, the knowledge of agents is a result of the outcome of empirical data analysis in addition to the pre-existing domain knowledge. BDPM of agents often involves detecting hidden patterns, constructing predictive and clustering models, identifying outliers, etc. This collective 'intelligence' of MAS must be developed by distributed domain knowledge and analysis of the distributed data observed by different agents. They collaborate to exchange knowledge that is extracted from data at different geographically distributed network nodes, creating as a result a distributed model of the environment – collective knowledge of the agents [28].

Usually, in MASs communication is expensive and not possible between all nodes. So usually BDPM for MAS suppose computations with minimum network communication and maximum local computations, if possible. Local computation is carried out on each node, and either a central node communicates with each distributed node to compute the global models or a peer-to-peer architecture is used. In case of the peer-to-peer architecture, individual nodes might communicate with a resource-rich centralised node, but they perform most tasks by communicating with neighbouring nodes through message passing over an asynchronous network [28]. In our works we demonstrated that properly established distributed BDPM may provide almost the same quality of the models as centralized traditional methods [8].

Following [31], the benefits of using MASs for BDPM are the following: 1) retaining the autonomy of data sources; 2) facilitating interactive BDPM; 3) improving dynamic selection of sources and data gathering; 4) providing high scalability to massive distributed data; 5) stimulating multi-strategy BDPM; 6) enabling collaborative BDPM.

3.3 Cloud Computing Infrastructure and BDPM

Architectures for cloud computing are characterized by close integration of its users to the system and creation of their virtual representations in terms of IoT. As there is Big Data associated with each user, and data is stored in the cloud, cloud computing systems automatically get distributed user Big Data. However, in contrast to the 'pure' MASs, the virtual users are fully connected and each piece of data can be accessed from other parts of the system.

The users of the cloud system usually have limited computational capacity because they are often connected to the cloud using mobile devices. So BDPM is performed partially locally, partially by other agents, which collect information from the users and store it in the cloud. Therefore the major issues of the BDPM are the workload and the communication costs. The implementation of the cloud computing handles a lot of this workload due to of the high connectivity of data agents/centres [32].

The main difference between BDPM for 'pure' MASs and BDPM for cloud computing is dealing with communication. In the first case communication is a bottleneck. In the second case communication is broadly available, the bottleneck is computation, because usually more information is available that can be processed. The BDPM methods with cloud computing infrastructure should know *which information* should be processed *and where* it is available. Here information *quality* should be taken into account as an important criterion.

BDPM remain distributed when applied in cloud computing infrastructure. However the question of the information availability is replaced by a question of the information cost, which takes such factors into account as the information location, speed of its extraction, quality, reliability, etc. BDPM methods in cloud computing deal with information, which is physically distributed, but is available subject to costs.

There are large numbers of users, which have similar but not equal requests to the cloud computing system. Therefore BDPM should be applied for the estimation of similar characteristics (e.g. travel time) based on the similar data in cloud computing system, however taking into account individual data and characteristics of a user (e.g. its vehicle type and current route information). For this purpose, different levels of data processing should be established as well as characteristics pre-calculated and then efficiently combined with actual user data.

The big challenge of the BDPM is parallelism, using computing power to gain precious time. The use of the cloud computing brings the ability of use many powerful interconnected servers with multi core processors without needing to implement it physically in every user's environment.

We see the following benefits of using BDPM in cloud environments: 1) virtual integration of data sources into system without physical integration; 2) facilitation of cost-based selective BDPM; 3) stimulation of multi-objective BDPM; 4) support for multi-stage BDPM and different levels of data processing.

Agent-based cloud computing paradigm combines the benefits of using MAS with cloud computing environment. BDPM obtains in this case all the advantages provided by each paradigm.

3.4 BDPM Trend: Computational Statistics

Cloud computing platform facilitates data collection and provides the necessary resources for the operation of the computationally intensive methods of computational statistics.

Computational statistics is the interface between statistics and computer science. It is the area of computational science specific to the mathematical science of statistics. It is aiming at the design of algorithms for implementing statistical methods on computers, including the ones unthinkable before the computer age (e.g. bootstrap, simulation), as well as to cope with analytically intractable problems [33].

Computational statistics supposes an application of iterative calculations instead of complex analytical models by using available data in different combinations. The resulting solution of the problem is approximate; however in many practical situations (big amount of available information, complex and hierarchical structure of analysed system, and dependency of data) this may give more robust and precise results as classical methods or even work in the situations where classical methods are not feasible

(e.g. Big Data). On the other hand, computational statistics application is simple and does not require complex analytical or symbolic procedures.

The term 'computational statistics' may also be used to refer to computationally intensive statistical methods including resampling methods, Markov chain Monte Carlo methods, local regression, kernel density estimation, generalized additive models as well as computational intelligence methods: artificial neural networks and genetic algorithms.

The methods of computational statistics are broadly applied for the mentioned in Section 3.1 statistical models. Resampling approach to regression parameter estimation within centralised architecture was considered in [34, 35, 36]. Genetic algorithms and classifier systems within centralised architecture were successfully applied for classification task solving (e.g., [37, 38].

In our BDPM methods we broadly apply methods of computational statistics such as resampling, bootstrap and kernel density for regression analysis and clustering within decentralised architecture [39, 40, 41].

3.5 Core Methods and Problems of BDPM in Transportation Networks

In contemporary ITS, modelling and forecasting of traffic flows is one of the important techniques that need to be developed [42]. It is an example of a complex stochastic system, in which many different factors should be estimated.

Due to the limitations of centralised approaches discussed above, decentralised MASs with autonomous agents allow vehicles to make decisions autonomously, which is fundamentally important for the representation of these networks [42].

We focus on the following problems of BDPM and demonstrate the advantages of decentralized architecture.

Travel time forecasting: Predictive models

Travel times play an important role in transportation and logistics. From travellers' viewpoints, the knowledge about travelling time helps to reduce delays and improves reliability through better selection of routes. In logistics, accurate travelling time estimation could help to reduce transport delivery costs and to increase the service quality of commercial delivery by bringing goods within the required time window. For traffic managers, travelling time is an important index for traffic system operation efficiency [43].

There are several studies in which a centralised approach is used to predict travel times. The approach was used in various ITS, such as in-vehicle route guidance and advanced traffic management systems. A good overview is given in [43]. To make the approach effective, agents should cooperate with each other to achieve their common goal via so-called gossiping scenarios. The estimation of the actual travelling time using vehicle-to-vehicle communication without MAS architecture was described in [44].

A combination of centralized and decentralized agent-based approaches to the traffic control was presented in [45]. In this approach, the agents maintain and share the 'local weights' for each link and turn, exchanging this information with a centralized traffic information centre. The decentralised MAS approach for urban traffic network was considered also in [46], where the authors forecast the traversal time for each link of the network separately. Two types of agents were used for vehicles and links, and a neural network was used as the forecasting model.

A promising approach to agent-based parameter estimation for partially heterogeneous data in sensor networks was suggested in [47]. Another decentralised approach for homogeneous data was suggested in [48] to estimate the parameters of a wireless network by using a parametric linear model and stochastic approximations.

A problem of decentralised travel time forecasting was considered in [19, 18, 8]. An MAS-based architecture with autonomous agents was implemented for this purpose. A decentralised linear [19, 8] and kernel density (KD) based [18, 8] multivariate regression models were developed to forecast the travelling time. The iterative least square estimation method was used for regression parameter estimation, which is suitable for streaming data processing. The resampling-based consensus method was suggested for coordinated adjustment of estimates between neighbouring agents. The efficiency of the suggested approach using simulation with data from the southern part of Hanover was illustrated. The experiments showed the efficiency of the proposed approach. The prediction technique in tutorial style was described in terms of distributed network intelligence in [8]. The comparison of parametric and non-parametric approaches for traffic-flow forecasting made in [49], demonstrates the efficiency of the non-parametric KD regression [20, 16].

Clustering traffic states

The most popular clustering and classification problems in traffic research are traffic state clustering [50] and participant behaviour clustering for group formation [51]. Clustering of travel-time information tried to discover homogeneous traffic patterns to be used with the common forecasting model. A KD clustering, which is a promising non-parametric method of computational statistics tool, allows arbitrary shaped clusters to be discovered. Fast clustering, which is based on KD, was described by Hinnenburg and Gabriel [52]. The distributed (with a central authority) version of KD-based clustering (KDEC scheme) was considered in [30]. Another decentralised graph-oriented not KD clustering approach was presented in [53].

A clustering problem with the application of the KD-based methods within a MAS architecture was considered in [54, 55]. The resampling-based consensus method was suggested for coordinated adjustment of estimates between neighbouring agents. The simulation experiments with real-world data were demonstrated.

Change point analysis

Change point analysis for detection of outliers in city traffic was investigated in [15, 56]. MAS architecture and implemented and two computational statistics-based resampling tests for change point detection were proposed at the BPMP layer of agent logics. The efficiency of the suggested approach was evaluated [56] for two different scenarios based on real traffic data [15]. In the first scenario, vehicles planed their routes based on the received information about the travel times. If a change point in travel times was detected, only the data after it was taken into account. In the second scenario, traffic light received the information about vehicle flows arriving to the intersection and applied one of the regulation strategies. If a change point in vehicle flow was detected, only the data after it was taken into account.

Traffic routing problem

A traffic routing problem with decentralized decision making of vehicle agents in urban traffic system was investigated, where the planning process for a vehicle agent is separated into two stages: strategic planning for selection of the optimal route and tactical planning for passing the current street in the optimal manner. A MAS architecture and the necessary computational statistics-based algorithms for comparing two routes in a

stochastic graph [20, 16], and the shortest path search were developed [57] which are carried out at strategic planning stage. The models were implemented to real data and integrated into a traffic domain application use case, where efficiency of the algorithms was evaluated. Distributed optimization approach for traffic flow routing was considered in [58].

■4.0 REFERENCE ARCHITECTURE FOR TRAFFIC BDPM

In the previous Section we demonstrated technologies and BDPM methods used in cloud-based ITS. Now we consider sample architecture of a cloud-based ITS and explain the main data flows that appear there. We show as well how the data flows can be processed using BDPM and provide sufficient information for fulfilling the user requests.

The applications executed in the cloud are data-intensive. Services provided through the cloud require large amounts of data to be processed, aggregated, and analysed. Then, the processed data is used for calculating optimal strategies for traffic participants.

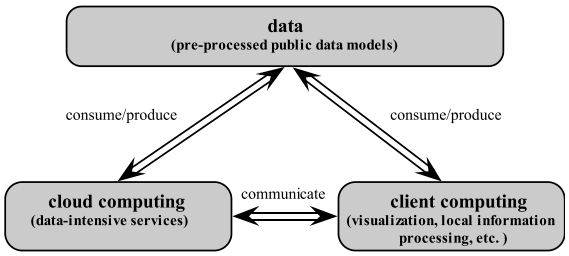


Figure 5 Data flow interconnections

As we already mentioned, computation is a bottleneck in cloud computing. So a very important challenge of BDPM is a reasonable processing balance between local data sources (clients) and a cloud. If a client has sufficient computational power, it can pre-process data locally and provide already processed data to the cloud, reducing cloud computations and network traffic. If however the computational power of a client does not allow information processing, the raw data are provided to the cloud and should be processed there. This is illustrated in Figure 5.

Now we consider reference architecture for BDPM and decision-making stages in ITS, which is described in our previous works [17] and based on [13]. In this architecture, we concentrate on illustrating data flows and their processing as well as using results for optimization of participant strategies and fulfilling their requests. A principal architecture of BDPM and decision-making stages in ITS is illustrated in Figure 6.

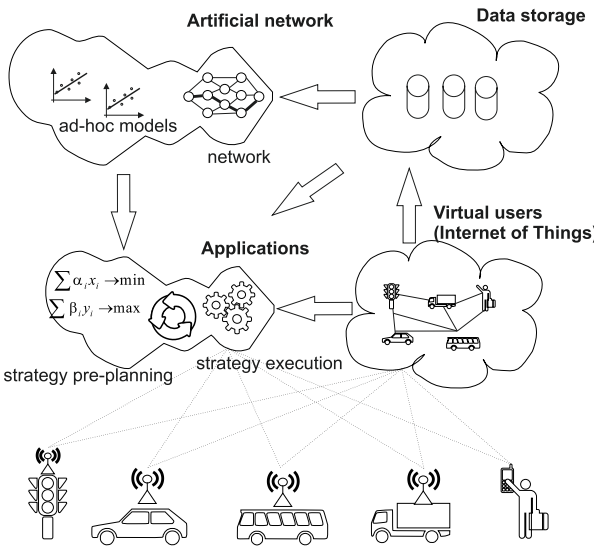


Figure 6 Reference architecture of BDPM and decision-making stages in ITS

It should be noted that usually many clouds from different providers are available. Some of the problems can be similar for them, and cooperation between them is possible.

The users of the ITS (traffic participants such as vehicles or pedestrians, business users such as logistic providers, public transports or taxis, data providers such as cameras or detectors, as well as traffic managers such as traffic management centres or traffic control elements) are connected with the cloud using stable and permanent Internet connection. This allows creating a virtual representation of each user in terms of IoT and having in the cloud dynamic sensor data, associated with them (pre-processed or raw). This creates a network of virtual users, which in fact is a mirror of reality in the cloud. This virtual reality contains distributed user data (partly stored in user devices, partly in the virtual storages provided by the cloud, but still associated with users). Note that disconnection of a user from the cloud does not mean elimination of its virtual representation; this only means that the locally stored data become unavailable in the cloud.

On the first stage data should be pre-processed. Raw sensor data requires very much storage space and cannot be stored for a long time. This data can be processed locally or upload to the cloud and pre-processed there. The results of the pre-processing are stored in the user profile and can be uploaded to the cloud at this stage.

The next stage is organizing of the virtual cloud information storages. This is made by cloud data mining agents, which collect the information, partially copying it to the storages in the cloud, partially making references to the user profiles, if they are available in the cloud. These agents put special attention to cost of the information, which includes its availability, reliability and precision. These virtual storages are subject of further BDPM.

Cloud-based systems have a big number of users, and should fast react to their requests. For this purpose artificial ad-hoc networks are created, which are oriented to concrete problems, solved by the cloud system. For example, the networks oriented to shortest path calculation, traffic light regulation or passenger transit can be created.

There are two important problems solved in the artificial network: estimation of its parameters and pre-calculation of user strategies.

Estimation of the ad-hoc network parameters is the main stage of BDPM. It consists in the estimation of the network parameters in order to obtain actual state of the network. Based on the information in the virtual storages estimation of the parameters is performed, taking costs of data into account and receiving data from physical storages if necessary. These parameters can be travel times on the network nodes, queues on the intersections or travel times between stops for public transport. A very important aspect is taking dynamic changes of the information into account, which is constantly provided by data mining agents.

Pre-calculation of the estimated parameters includes the following stages of data processing and network optimization:

Stage 1: Mining data from the IoT and its pre-processing. All the participants of the cloud-based system have virtual representations as active IoT components (agents). These virtual agents are associated with data (mostly real-time) and act as data sources for the cloud-based system. The cloud system locates and collects the necessary data from different agents, and provides usual data mining operations (changes and outliers are detected, preliminary aggregation and dimensionality reduction are performed). The collected data are stored as historical information in the cloud and are used later as input data for ad-hoc network models (Stage 2). Stream-based methods of semi-decentralized change-point detection, outlier detection, clustering and classification, and factor analysis occur regularly in this stage.

Stage 2: Ad-hoc network models. The application-specific digital networks of virtual traffic participants (e.g. regional, social) are created, and the corresponding data models are used in order to estimate the important characteristics and parameters of these networks using the information collected in Stage 1 and for strategy optimization at Stage 3. The future behaviour of traffic participants is forecasted as well. Semi-decentralized, flows forecasting (possibly with incomplete information) methods such as (multiple-response) regression models, Bayesian networks, time series, simulation, are also applied at this stage. Many pre-defined data models can run concurrently in the digital network. The corresponding data storages are located in the cloud and are semi-centralized, so the methods should take costs of different pieces of information into account.

Stage 3: Static decisions and initial strategy optimization. Cloud applications use pre-calculated results of the ad-hoc network models from Stage 2 and the available historical information (including private information) about the traffic network to perform their pre-planning tasks. Initial optimization of the strategies is resource expensive, and can be partially pre-calculated in ad-hoc network models and then instantiated according to the application's goals and preferences. These models are also checked in the digital traffic network. This stage can require aggregation of different data models and existing strategies. Methods of self-learning stochastic (multi-criteria) optimization such as neural networks, decision trees, Markov decision processes, choice models, graph optimization algorithms are used.

Stage 4: Dynamic decisions and strategy update. The pre-planned tasks from Stage 3 are executed, and updates are made according to the dynamic real-time situation extracted from the virtual agents. The aggregation of the pre-planned data and strategies with the dynamic ones is the most important problem at this stage. An additional difficulty here is the requirement of fast real-time execution. (Automatic) cooperation between users in their decisions is possible; therefore, stream-based methods of data models and strategy updates such as reinforcement learning, Bayesian networks, dynamic decision trees, stream regression, and distributed constraint satisfaction/optimization can be applied.

5.0 BDPM AND DECISION-MAKING STAGES IN in ITS SCENARIOS

We consider three ITS application scenarios: 1) A cooperative intersection control, which optimizes vehicle flows in traffic networks by regulating the intersection controllers. 2) A personal travel companion, which provides dynamic planning and monitoring of multimodal journeys to travellers, surface vehicle drivers, and transports operators. 3) A logistics services companion, which provides benefits to clients and stakeholders involved in, affected by, or dependent on the transportation of goods in urban environments. We demonstrate the most important stages of BDPM and optimization in order to derive requirements for a general architecture described in the previous section.

5.1 Virtualized Cooperative Intersection Control

This scenario uses adaptive, semi-distributed traffic management strategies hosted in the cloud for the regulation of intersection controllers, and creates ad-hoc networks in the cloud between clusters of vehicles and the traffic management infrastructure. It recommends the optimal speed to drivers to keep the traffic flow smooth, and assists adapting traffic controllers (e.g. traffic lights, signs) based on the real time traffic situation. This service uses real-time traffic information and a route-data collection service to formulate strategies for the optimization of network operation.

Stage 1: Processing the following data streams (historical and real-time): 1) floating-car data (speeds, positions, etc.); 2) sensor data from the infrastructure (loops, traffic lights, etc.); 3) information about routes and actual locations of collective transport (public transport, taxi, shared cars, etc.) 4) data from distribution vehicles (logistic transport); 5) weather conditions; 6) accidents, car breakdowns, road-works; 7) organizational activities (sport events, conferences, etc.)

Stage 2: Creating ad-hoc networks, which are virtual abstract networks for solving specific problems (intersection and regional traffic models, green wave models, public transport priority, jam avoidance, etc.). Estimating network parameters (traffic flux, density, and speed, travel time estimation, etc.).

Stage 3: Developing static strategies of intersection control and cooperation (such as increase of flows, changed weather conditions, organizational activities); cooperation plans of clusters of vehicles, etc.).

Stage 4: Combining dynamic real-time information with static strategies in order to receive up-to-date controlling decisions (correction of signal plans according to current conditions, cooperation of signal controllers to resolve problems such as jams, accidents, etc.) based on historical information, previous experience, and data models from the previous stage (traffic light signal plan optimization; signal plans for expected events.

5.2 Dynamic Multi-modal Journey Planning

This scenario helps travellers plan and adjust a multi-modal, door-to-door journey in real-time. It provides improved (i.e., quicker, more comfortable, cheaper, and greener) mobility to daily commuters and other travellers by identifying optimal transportation means and a strong real-time orientation. This planning proposal for a multi-modal journey takes into account the current means of transportation, the traveller's context and preferences, city traffic rules, and the current requirements and constraints. The journey plan needs to obtain an overall indication of the trip duration as well as accommodate early reservation of resources (train or plane ticket).

Stage 1: Processing of the following data streams (historical and dynamic) in addition to the previous application: 1) floating passenger data; 2) travellers' preferences; 3) timetables and availability of collective transport (tickets, shared cars availability, etc.); 4) changes in time-tables.

Stage 2: Creation of ad-hoc networks (transit stations, public transport coordination, passenger choice of transport, etc.) and estimation of network parameters (travel time for different transport modes depending on various factors, waiting times, passenger arrival at stops, price models, etc.).

Stage 3: Multi-modal route pre-planning based on historical data and estimated network parameters for expected conditions (pre-planning for popular routes, preplanning for pre-booked routes, pre-planning for expected events) as well as optimal time-table calculation for public transport based on the expected conditions.

Stage 4: Dynamic update of pre-planned routes for the actual multi-modal journey (actual travel-time estimation, re-planning in the case of delays in previous trips in the multi-modal chain, re-planning for additional travel possibilities, or cancelling a part of the multi-modal journey), as well as dynamic update of public transport time-tables (on-demand changes, co-ordination of different transport means).

5.3 Itinerary Booking and Real-time Optimized Route navigation

This scenario helps a logistics provider (1) to guarantee quick (especially on-time) deliveries at a low cost based on up-to-date information and (2) to maximize the efficiency of each vehicle and the fleet. It is fundamental to optimize the movements of the logistics vehicles, to help them avoid traffic jams and take the shortest routes when possible.

Stage 1: Processing of the following data streams (historical and dynamic) in addition to the first application: 1) order data (transportation demand); 2) available logistic vehicles (possible load, speed, etc.); 3) timetables (if necessary) and actual positions of the vehicles; 4) client data (drop-off preferences, actual location, etc.).

Stage 2: Creation of ad-hoc networks (delivery models, logistic provider-client interaction models, etc.), and estimation of the network parameters (travel times for different route segments, delay probability, drop-off process time distribution, probability of accidents, probability of problems with vehicles, etc.).

Stage 3: Pre-planning of the delivery process (preliminary good distribution by vehicles, preliminary order of clients for each vehicle, preliminary route for each vehicle, preliminary time window for each client, etc.). Note that the itineraries of large logistic operators can be used to provide better predictions of the traffic situation using virtualized cooperative intersection intelligence application as well as by applying priority rules for logistic vehicles during booking.

Stage 4: Dynamic update of pre-planned delivery routes depending on up-to-date information (re-planning of routes depending on current traffic situation, re-planning in the case of accidents or traffic jams, re-planning in the case of vehicle problems, estimation of actual delivery time, etc.). Cooperation between logistic vehicles (exchange or orders, adoption of other vehicle's orders in the case of problems, etc.). Dynamic agreement with clients (agreement about drop-off place depending on current position of the vehicle and client, agreement about change of drop-off time, reaction to the new/changed customer requests, etc.).

6.0 DECENTRALISED BDPM METHODS

We provide an overview of two very important groups of BDPM methods, which can be applied for complex stochastic systems represented with MAS architecture using the power of cloud-based environment. *Decentralised clustering* groups together similar data, reducing Big Data dimensions [54, 55]. Decentralized data clustering and the corresponding classification can be considered as a prerequisite step to implementation of forecasting and decision-making models.

Decentralised regression provides estimation of some characteristics depending on given values of factors [19, 18, 8]. Both methods are distributed, which allows combination of local and global computations, which provides a very good basis for their application in cloud context.

Decentralized clustering is usually applied at early stages of data processing (e.g. at Stage 1 and early Stage 2 of the architectures described in Section 4). It can be used, e.g., to group together: 1) travel-time data or similar travel segments in the dynamic journey planning scenario; 2) travellers' behaviour and traffic flows under similar conditions in the virtualized intersection control; 3) similar orders and client locations in the itinerary booking scenario. Different regression models are then applied for different clusters.

Decentralized regression is mostly used in ad-hoc data models, which correspond to Stage 2 of the architecture from Section 4. Regression models can be used for travel time prediction in the dynamic journey planning scenario, traffic flows prediction in the virtualized intersection control or order appearance prediction in the itinerary booking scenario.

6.1 Decentralised Regression

Regression analysis models the dependency between the dependent variable (usually denoted by $\mathbf{Y}$) and independent variables (factors, usually denoted by $\mathbf{X}$). It describes the dependence of conditional expectation $E[\boldsymbol{Y}|\boldsymbol{X}]$ on the values of several independent variables (factors) $\mathbf{X}$.

A *linear regression* model supposes that the dependent variable $\mathbf{Y}$ is a linear combination of the factors $\mathbf{X}$. The linear regression model [21] in the matrix form can be expressed as

$$\mathbf{Y} = \mathbf{X}\boldsymbol{\beta} + \boldsymbol{\varepsilon}, \tag{6.1}$$

where n is a number of observations, d is a number of factors, $\mathbf{Y}$ is an $n \times 1$ vector of dependent variables; $\mathbf{X}$ is an $n \times d$ matrix of explanatory (dependent) variables; $\boldsymbol{\beta}$ is an $d \times 1$ vector of unknown coefficients, which are parameters of the system to be estimated; $\boldsymbol{\varepsilon}$ is an $n \times 1$ vector of random errors. The rows of the matrix $\mathbf{X}$ correspond to observations and the columns correspond to factors.

The well-known least square estimator $\mathbf{b}$ of $\boldsymbol{\beta}$ is [21]:

$$\mathbf{b} = (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{Y}. \tag{6.2}$$

We can make a forecast for a certain $n+1$-th future values of the factors $\mathbf{x}_{n+1}$ as follows:

$$E[Y_{n+1}] = \mathbf{x}_{n+1}\mathbf{b}. \tag{6.3}$$

A *KD regression* model does not make certain assumptions about the form of dependence of dependent variable $\mathbf{Y}$ on factors $\mathbf{X}$. It can predict observations without reference to a fixed parametric model. In contrast, it uses kernel as a weighting function in the estimation process.

A general form of a non-parametric regression model is [59]:

$$\mathbf{Y} = m(\mathbf{x}) + \boldsymbol{\varepsilon}, \tag{6.4}$$

where $\boldsymbol{\varepsilon}$ is a random error such that $\mathrm{E}[\boldsymbol{\varepsilon} \mid \mathbf{X} = \mathbf{x}] = \mathbf{0}$ and $\mathrm{Var}[\boldsymbol{\varepsilon} \mid \mathbf{X} = \mathbf{x}] = \sigma^2(\mathbf{x})$; and $m(\mathbf{x}) = \mathrm{E}[\mathbf{Y} \mid \mathbf{X} = \mathbf{x}]$.

The Nadaraya-Watson KD estimator [59] is

$$\widehat{m}_n(\mathbf{x}) = \frac{\sum_{i=1}^{n} H^{-1}K(\mathbf{x}-\mathbf{X}_i)\,Y_i}{\sum_{i=1}^{n} H^{-1}K(\mathbf{x}-\mathbf{X}_i)} = \frac{\sum_{i=1}^{n} \omega_i(\mathbf{x})\,Y_i}{\sum_{i=1}^{n} \omega_i} == \sum_{i=1}^{n} \widehat{\omega}_i(\mathbf{x}) Y_i, \tag{6.5}$$

where $K(\bullet)$ is the kernel function of R^d, $\mathbf{H}$ is the bandwidth matrix, $\omega_i(\mathbf{x}) = H^{-1}K(\mathbf{x}-\mathbf{X}_i)$ are the weights of the historical observations in the forecast of $\mathbf{x}$ and , $\widehat{\omega}_i(\mathbf{x}) = \frac{\omega_i(\mathbf{x})}{\sum_{i=1}^{n}\omega_i}$ are the corresponding normalised weights. If there is no additional information about factor dependence $\mathbf{H} = diag(h_1, h_2, \ldots, h_d)$. Kernel function is a symmetric density function, for example, a standard multivariate normal distribution $\mathbb{N}(0, I)$, called the multivariate Gaussian kernel.

Using the Nadaraya-Watson estimator, a forecast for any future factors values $\mathbf{x}_{n+1}$ is:

$$E[Y_{n+1}] = \widehat{m}_n(\mathbf{x}_{n+1}). \tag{6.6}$$

Now we consider *distributed versions* of the above-mentioned linear [19, 8] and KD [18, 8] regressions. We use MAS architecture, where each agent makes estimations of its models locally based on its observations. If an agent experience difficulties with any new prediction, it makes help-request to other agents and receives responses as the model parameters in the case of linear regression or as raw data in the case of KD regression. These responses are used to improve initial model of the requested agent.

Suppose we have a MAS consisting of s autonomous agents $\boldsymbol{Ag} = \{Ag^{(1)}, Ag^{(2)}, \ldots, Ag^{(s)}\}$, and each of them contains a local regression model, which is estimated on the basis of its experience.

Let us describe two versions (linear and KD regression) of parameter adjustment algorithms, which allow the agents to exchange the information in order to improve the forecasts.

Consider an agent $Ag^{(i)}$ which makes a forecasting for some factors $\mathbf{x}_{n+1}^{(i)}$ using one of the regression models. After forecasting, $Ag^{(i)}$ checks whether a forecast $E\left[Y_{n+1}^{(i)}\right]$ is not reliable and it needs help from other agents.

For the linear model, an approach to check a reliability of a forecast is to compare the width of the confidence interval of the forecast with a forecast value. The forecast $E\left[Y_{n+1}^{(i)}\right]$ is considered to be not reliable if its value is sufficiently smaller than its confidence interval width:

$$\frac{\Delta^{(i)}(\mathbf{x}_{n+1}^{(i)})}{\mathbf{x}_{n+1}^{(i)}\mathbf{b}^{(i)}} > p,$$

where p is an agent's parameter representing the maximal ratio of the confidence interval width to the forecast, after which a coordination takes place; $\Delta^{(i)}(\mathbf{x}_{n+1}^{(i)})$ is the confidence interval for factors $\mathbf{x}_{n+1}^{(i)}$ based on the agent's data.

For the KD model, we consider a forecast for $\mathbf{x}_{n+1}^{(i)}$ as not reliable if only one of the observations is taken with a significant weight in this forecast:

$$\max\left(\widehat{\omega}_c^{(i)}\left(\mathbf{x}_{n+1}^{(i)}\right)\right) > bp,$$

where bp is an agent's parameter representing the maximal weight, after which a coordination takes place.

The parameter adjustment procedure is the following. First, agent $Ag^{(i)}$ sends a request, containing a set of factors $\mathbf{x}_{n+1}^{(i)}$ as well as a threshold $Tr^{(i)}\left(\mathbf{x}_{n+1}^{(i)}\right)$ to other agents within its transmission radius.

For the linear model, this threshold is set equal to the confidence interval width: $Tr^{(i)}\left(\mathbf{x}_{n+1}^{(i)}\right) = \Delta^{(i)}\left(\mathbf{x}_{n+1}^{(i)}\right)$.

For the KD model, this threshold is set as the weight of the second best observation: $Tr^{(i)}\left(\mathbf{x}_{n+1}^{(i)}\right) = \omega_{(n-1)}^{(i)}\left(\mathbf{x}_{n+1}^{(i)}\right)$, where an ordered sequence of weights in the forecast for $\mathbf{x}_{t+1}^{(i)}$ is represented by $\omega_{(1)}^{(i)}\left(\mathbf{x}_{n+1}^{(i)}\right), \omega_{(2)}^{(i)}\left(\mathbf{x}_{n+1}^{(i)}\right), \ldots, \omega_{(n)}^{(i)}\left(\mathbf{x}_{n+1}^{(i)}\right)$.

Each agent $Ag^{(j)}$ in the transmission radius makes the following after the receiving of data.

For the linear model, it calculates the confidence interval $\Delta^{(j)}(\mathbf{x}_{n+1}^{(i)})$ for the requested values of factors $\mathbf{x}_{n+1}^{(i)}$ on the basis of its data and compares it with the threshold. If this value is less than the threshold, $\Delta^{(j)}\left(\mathbf{x}_{t+1}^{(i)}\right) < Tr^{(i)}\left(\mathbf{x}_{t+1}^{(i)}\right)$, $Ag^{(j)}$ replies to $Ag^{(i)}$ by sending its parameters $\mathbf{b}^{(i)}$.

For the KD model, it calculates the weights $\omega_c^{(j)}\left(\mathbf{x}_{n+1}^{(i)}\right)$ on the basis of its own data. If there are observations with weights $\omega_c^{(j)}\left(\mathbf{x}_{n+1}^{(i)}\right) > Tr^{(i)}\left(\mathbf{x}_{n+1}^{(i)}\right)$ it forms a reply $\widehat{D}^{(j,i)}$ from these observations (maximum 2) and sends it to $Ag^{(i)}$.

Let $\boldsymbol{G}^{(i)} \subset \boldsymbol{Ag}$ be a group of agents, who are able to reply to $Ag^{(i)}$ by sending the requested data.

For the linear model, each $Ag^{(j)} \in \boldsymbol{G}^{(i)}$ sends its parameters $\mathbf{b}^{(j)}$. $Ag^{(i)}$ receives replies from the group $G^{(i)}$. It assigns weights to each $Ag^{(j)}$ (including itself) $c^{(i)}=\left\{c_j^{(i)}\right\}$. These weights are time-varying and represent the reliability level of each $Ag^{(j)}$, including reliability of own experience. In our case, the agents' weights depend on the forecasting experience. We assume that $c^{(i)}$ is a stochastic vector for all n , i.e. the sum of its elements is equal to 1. Then an updated estimate is

$$\tilde{\mathbf{b}}^{(i)} = \sum_{Ag^{(j)} \in G^{(i)}} c_j^{(i)}\,\mathbf{b}^{(j)}. \tag{6.7}$$

For the KD model, the agents $Ag^{(j)}$ send their data, which are closest to the requested point. All the data $\widehat{D}^{(j,i)}, Ag^{(j)} \in \boldsymbol{G}^{(i)}$ received by $Ag^{(i)}$ are verified and duplicated data are discarded. These new observations are added to the dataset of $Ag^{(i)}$: $D^{(j)} \leftarrow \cup_{Ag^{(j)} \in G^{(i)}} \widehat{D}^{(j,i)} \cup D^{(j)}$. Suppose that $Ag^{(i)}$ received r observations. Then, the new KD function of $Ag^{(i)}$ is updated by considering the additive nature of this function:

$$\widetilde{m}_{n+r}^{(i)}(\mathbf{x}) = \frac{p_n^{(i)}(\mathbf{x}) + \sum_{Ag^{(j)} \in \boldsymbol{G}^{(i)}} p^{(i,j)}(\mathbf{x})}{q_n^{(i)}(\mathbf{x}) + \sum_{Ag^{(j)} \in \boldsymbol{G}^{(i)}} q^{(i,j)}(\mathbf{x})}, \tag{6.8}$$

where $p^{(i,j)}(\mathbf{x})$ and $q^{(i,j)}(\mathbf{x})$ are the nominator and denominator of (6.5), respectively, calculated by $\widehat{D}^{(j,i)}$. Finally, $Ag^{(i)}$ can autonomously make its forecast for $\mathbf{x}_{n+1}^{(i)}$ as $\widetilde{m}_{n+r}^{(i)}(\mathbf{x}_{n+1}^{(i)})$.

6.2 Decentralised Clustering

We formulate first the clustering problem and describe the KD clustering algorithm [54, 55]. Let $\boldsymbol{X} = \{\boldsymbol{x}_1, \boldsymbol{x}_2, \dots, \boldsymbol{x}_N\}$, $\boldsymbol{x}_i \in R^d$ be a dataset to be clustered into k non-overlapping subsets $\{S_1, S_2, \dots, S_k\}$.

Non-parametric clustering methods [30] are well suited for exploring clusters without building a generative model of the data. KD clustering consists of a two-step procedure: estimation and optimisation. During the estimation step, the probability density of the data space is directly estimated from data instances. During the optimisation step, a search is performed for densely populated regions in the estimated probability density function.

Estimation step. The density function is estimated by defining the density at any data object as being proportional to a weighted sum of all objects in the data-set, where the weights are defined by an appropriately chosen kernel function [30].

A KD estimator is

$$\hat{\psi}^{[X]}(\mathbf{x}) = \frac{1}{N}\sum_{\mathbf{x}_i \in \mathbf{X}} K_{\mathbf{H}}(\|\mathbf{x} - \mathbf{x}_i\|), \qquad (6.8)$$

where $\|\mathbf{x} - \mathbf{x}_i\|$ is a distance between $\mathbf{x}_i$ and $\mathbf{x}$, $K_H(\bullet) = |\mathbf{H}|^{-1}K(\mathbf{H}^{-1}\bullet)$[42]. We use the multivariate Gaussian kernel function in our study: $K(\mathbf{x}) = (2\pi)^{-d/2}\exp(-\frac{1}{2}\mathbf{x}^T\mathbf{x})$.

Optimisation step. Maxima of KD are detected and groups all of the data objects in their neighbourhood are grouped into corresponding clusters. We use a hill climbing method for KD maxima estimation with Gaussian kernels (DENCLUE) [52] and modify the technique for the multivariate case. This method converges towards a local maximum and adjusts the step size automatically at no additional costs. Other optimization methods [52] require more steps and additional computations for step size detection. Each KD maximum can be considered as the centre of a point cluster. With centre-defined clusters, every local maximum of $\hat{\psi}(\bullet)$ corresponds to a cluster that includes all data objects that can be connected to the maximum by a continuous, uphill path in the function of $\hat{\psi}(\bullet)$. Such centre-defined clusters allows for arbitrary-shaped clusters to be detected, including non-linear clusters. An arbitrary-shape cluster is the union of centre-defined clusters that have maxima that can be connected by a continuous, uphill path. The goal of the hill climbing procedure is to maximize the KD $\hat{\psi}^{[X]}(\mathbf{x})$. By setting the gradient $\nabla\hat{\psi}^{[X]}(\mathbf{x})$ of KD to zero and solving the equation $\nabla\hat{\psi}^{[X]}(\mathbf{x}) = \mathbf{0}$ for $\mathbf{x}$, we get:

$$\mathbf{x}^{(l+1)} = \frac{\sum_{\mathbf{x}_i \in \mathbf{X}} K_{\mathbf{H}}(\|\mathbf{x}^{(l)} - \mathbf{x}_i\|)\mathbf{x}_i}{\sum_{\mathbf{x}_i \in \boldsymbol{X}} K_{\mathbf{H}}(\|\mathbf{x}^{(l)} - \mathbf{x}_i\|)}, \qquad (6.9)$$

The formula (6.9) can be interpreted as a normalized and weighted average of the data points. The weights for each data point depend on the influence of the corresponding kernels on $\mathbf{x}^{(l)}$. Hill climbing is initiated at each data point $\mathbf{x}_i \in \mathbf{X}$ and is iterated until the density does not change, i.e. $\left[\hat{\psi}^{[X]}\left(\mathbf{x}_i^{(l)}\right) - \hat{\psi}^{[X]}\left(\mathbf{x}_i^{(l-1)}\right)\right] / \hat{\psi}^{[X]}\left(\mathbf{x}_i^{(l)}\right) \le \boldsymbol{\epsilon}$, where $\boldsymbol{\epsilon}$ is a small constant. The end point of the hill climbing algorithm is denoted by $\mathbf{x}_i^* = \mathbf{x}_i^{(l)}$, corresponding to a local maximum of KD.

Now we should determine a cluster for $\mathbf{x}_i$. Let $\mathbf{X}^c = \{\mathbf{x}_1^c, \mathbf{x}_2^c, \dots\}$ be an ordered set of already identified cluster centres (initially, we suppose $\mathbf{X}^c \neq \emptyset$). We find an index of the nearest cluster centre from $\mathbf{x}_i^*$ in the set $\mathbf{X}^c$:

$$nc(\mathbf{x}_i^*) = \arg\min_{\mathbf{x}_j^c \in \mathbf{X}^c} \left\|\mathbf{x}_j^c - \mathbf{x}_i^*\right\|.$$

We describe now the cooperation for sharing the clustering experience among the agents in a network. We introduce the following definitions. Let $\boldsymbol{Ag} = \{Ag^{(1)}, Ag^{(2)}, \dots, Ag^{(s)}\}$, be a group of s agents. Each $Ag^{(j)}$ has a local dataset $D^{(j)} = \left\{\mathbf{x}_n^{(j)} | n = 1, \dots, N^j\right\}$, where $\boldsymbol{x}_n^{(j)} \in R^d$. In order to underline the dependence of the KD function (6.8) on the local dataset of $Ag^{(j)}$, we denote the KD function by $\hat{\psi}^{[\boldsymbol{D}^{(j)}]}(\mathbf{x})$.

Consider a case when some agent $Ag^{(i)}$ is unable to classify (after optimization has formed a new or small cluster) some future data point $\mathbf{x}_{n+1}^{(i)}$ because it does not have sufficient data in the neighbourhood of this point. Our model uses a two-phase protocol for performing communication between agents. First, $Ag^{(i)}$ sends the data point $\mathbf{x}_{n+1}^{(i)}$ to the other neighbouring agents.

We consider two different helping procedures: non-parametric, in which some data is transmitted without any parameter estimation and semi-parametric, where the observations are approximated with the mixture of multi-dimensional Gaussian distributions and their parameters are transmitted. Let $\boldsymbol{G}^{(i)} \subset \boldsymbol{Ag}$ be a group of agents, who are able to reply to $Ag^{(i)}$ as it was described in section 6.1

In the non-parametric procedure, considered in [55], each $Ag^{(j)}$ classifies $\mathbf{x}_{n+1}^{(i)}$ using its own KD function $\hat{\psi}^{[\boldsymbol{D}^j]}\left(\mathbf{x}_{n+1}^i\right)$ and performs the optimization step to identify the cluster for this point. Let $n_{j,i}$ be a number of points in the cluster of $\mathbf{x}_{n+1}^{(i)}$, not including $\mathbf{x}_{n+1}^{(i)}$ itself. In the case of successful clustering ($n_{j,i} > 0$), $Ag^{(j)} \in \boldsymbol{G}^{(i)}$ forms an answer $\boldsymbol{D}^{(j,i)}$ with c nearest points to the requested data point from the same cluster as $\mathbf{x}_{n+1}^{(i)}$, (or all points from the cluster, if $n_{j,i} \le$ c). Let $c_{j,i}$ be a number of points in the response $\boldsymbol{D}^{(j,i)}$. Each agent $Ag^{(j)} \in \boldsymbol{G}^{(i)}$ sends $\boldsymbol{D}^{(j,i)}$ together with $c_{j,i}$ and $n_{j,i}$ to $Ag^{(i)}$. After receiving all the answers, $Ag^{(i)}$ forms a new dataset $\widehat{\boldsymbol{D}}^{(j,i)}$.

In the semi-parametric procedure [54], in response to the help-request, $Ag^{(j)}$ send parameters from their estimated KD functions. Since the KD function is non-parametric and estimated directly from observations, we approximate the function with a mixture of multi-dimensional Gaussian distributions. Each $Ag^{(j)}$ identifies cluster associated with point $\mathbf{x}_{n+1}^{(i)}$ and performs the approximation of clusters with a mixture of normal distributions. Next, $Ag^{(j)}$ transmits the cluster parameters (weight, mean and covariance matrix). The agent $Ag^{(i)}$ adds this information to its KD and updates its clusters. Since parameter transmission requires less data, this approach requires less transmission, however, the approximation reduces the cluster shapes to a union of ellipsoids.

Let us consider an approximation step that approximates KD function with a mixture of distributions. This can be achieved with the expectation maximisation (EM) algorithm proposed by Dempster [60]. The approach is widely used for calculation of the maximum likelihood estimate of mixture models. In a mixture model, the probability density function is

$$f(\mathbf{x}; \Theta) = \sum_{b=1}^{B} \pi_b f_b(\mathbf{x}; \Theta_b),$$

where π_b are positive mixing weights that sum to one, f_b are component density functions parameterized by Θ_b, and $\Theta = \{\pi_b, \Theta_b\}$ are the model parameters. Each observation is assumed to be from one of the B components. A common choice for

component density is a multivariate normal distribution with parameters $\Theta_b = (\mu_b, \Sigma_b)$, where μ_b is a mean and Σ_b is a covariance matrix.

In the EM procedure, the expected likelihood of a given data-set is iteratively maximized. The algorithm [60] alternates between the E and the M steps until convergence is achieved.

We assume that each helping-agent $Ag^{(j)} \in \boldsymbol{G}^{(i)}$ receives data point $\mathbf{x}_{n+1}^{(i)}$, and tries to classify it. If it is successful, of $Ag^{(j)}$ determines that $\mathbf{x}_{n+1}^{(i)}$, belongs to a specific cluster and executes the EM-algorithm with this cluster. This algorithm approximates the cluster using a mixture of B^j multidimensional normal distributions with parameters $\Theta^j = \{\mu^j, \Sigma^j, \pi^j\}$, where $\mu^j = \{\mu_1^j, \dots, \mu_{B^j}^j\}$, $\Sigma^j = \{\Sigma_1^j, \dots, \Sigma_{B^j}^j\}$, and $\pi^j = \{\pi_1^j, \dots, \pi_{B^j}^j\}$, which are then returned to $Ag^{(i)}$. After receiving all answers, the agent $Ag^{(i)}$ has a vector of the parameters $\{\Theta^j\}$. The answers $\{\Theta^j\}$ can be interpreted by the agent $Ag^{(i)}$ as data points μ^j with the only difference being that the additional weights π^j and bandwidths from Σ^j should now be taken into account.

The next problem is the updating of the KD function of $Ag^{(i)}$. Denote $\widehat{\boldsymbol{D}}^{(i)}$ as a dataset of the agent of $Ag^{(i)}$ that includes the received answers. We take into account that density estimates (6.8) of each agent are additive, i.e. the aggregated density estimate $\hat{\psi}^{[\boldsymbol{D}^{(j)}]}(\mathbf{x})$ can be decomposed into the sum of the local density estimates.

For the non-parametric approach, $\hat{\psi}^{[\boldsymbol{D}^{(j)}]}(\mathbf{x})$ is updated with respect to the new knowledge $\boldsymbol{D}^{(j,i)}$ with one estimate for every dataset $\boldsymbol{D}^{(j,i)}$:

$$\hat{\psi}^{[\widehat{\boldsymbol{D}}^{(i)}]}(\mathbf{x}) = w_i \hat{\psi}^{[\boldsymbol{D}^{(i)}]}(\mathbf{x}) + \\ + \frac{(1-w_i)}{\sum_{Ag^{(j)} \in \boldsymbol{G}^{(i)}} n_{j,i}} \sum_{Ag^{(j)} \in \boldsymbol{G}^{(i)}} n_{j,i}\, \hat{\psi}^{[\boldsymbol{D}^{(j,i)}]}(\mathbf{x}), \qquad (6.10)$$

where w_i is a weight used for the agent's own local observations.

For the semi-parametric procedure, it uses the estimated parameters

$$\hat{\psi}^{[\widehat{\boldsymbol{D}}^{(i)}]}(\mathbf{x}) = w_i \hat{\psi}^{[\boldsymbol{D}^{(i)}]}(\mathbf{x}) + \\ + \frac{(1-w_i)}{\sum_{Ag^{(j)} \in \boldsymbol{G}^{(i)}} \pi_b^j \sum_{Ag^{(j)} \in \boldsymbol{G}^{(i)}} B^j} \sum_{\substack{Ag^{(j)} \in \boldsymbol{G}^{(i)} \\ b \in B^j}} \pi_b^j K_{\Sigma_b^j}\left(\|\mathbf{x} - \mu_b^j\|\right). \qquad (6.11)$$

After updating its KD function, $Ag^{(i)}$ performs a hill-climbing optimization procedure to identify clusters in its local data space.

To measure the clustering similarity [61] among the agents $Ag^{(i)} \in \boldsymbol{Ag}$ we use the following representation of a class labelling by a matrix $\mathbf{C}$ with components:

$$\mathbf{C}_{i,j} = \begin{cases} 1 \text{ if } \mathbf{x}_i \text{ and } \mathbf{x}_j \text{ belong to the same cluster and } i \neq j, \\ 0 \text{ otherwise} \end{cases}$$

Let two labelling have matrix representations $\mathbf{C}^{(1)}$ and $\mathbf{C}^{(2)}$, respectively. We define a dot product that computes the number of pairs clustered together $\langle \mathbf{C}^{(1)}, \mathbf{C}^{(2)} \rangle = \sum_{i,j} \mathbf{C}_{i,j}^{(1)} \mathbf{C}_{i,j}^{(2)}$. The Jaccard's similarity measure can be expressed as

$$J\left(\mathbf{C}^{(1)}, \mathbf{C}^{(2)}\right) = \frac{\langle \mathbf{C}^{(1)}, \mathbf{C}^{(2)} \rangle}{\langle \mathbf{C}^{(1)}, \mathbf{C}^{(1)} \rangle + \langle \mathbf{C}^{(2)}, \mathbf{C}^{(2)} \rangle - \langle \mathbf{C}^{(1)}, \mathbf{C}^{(2)} \rangle}.$$

7.0 EXPERIMENTAL RESULTS

In this section, we evaluate the efficiency of the above described decentralized regression and decentralized clustering using transportation application domain. Some numerical experiments illustrate quality of considered BDPM methods.

We consider transportation network data from the southern part of Hanover (Germany). The network (Figure 7) contained three parallel and five perpendicular streets, which formed 15 intersections with a flow of approximately 5000 vehicles per hour.

Figure 7 Road network in the southern part of Hanover, Germany

Table 1 Available factors of transportation data

Variable	Description
Y	travelling time (min);
$X^{\langle 1 \rangle}$	route length (km)
$X^{\langle 2 \rangle}$	average speed in system (km/h)
$X^{\langle 3 \rangle}$	average number of stops(units/min)
$X^{\langle 4 \rangle}$	congestion level (vehicles/h)
$X^{\langle 5 \rangle}$	number of the traffic lights in the route (units)
$X^{\langle 6 \rangle}$	number of the left turns in the route (units)

Available data (Table 1) consists of a dependent variable Y (travelling time) and six factors $\mathbf{X}^{\langle 1 \rangle}$ - $\mathbf{X}^{\langle 6 \rangle}$.

7.1 Decentralised Regression

We considered travelling time forecasting using linear and KD regression models [8]. We compared the following four estimates:

1) Linear estimate was produced by linear regression model.
2) Kernel density (KD) estimate was produced by KD regression model.
3) 'Oracle' estimate, which is a combination of linear and KD estimates. This estimate presented the optimum and helped to select the best forecast (with a low forecasting error) for every forecast. It is the best forecast that could be achieved using the linear and KD regressions.
4) 'Average' estimate was the average of the KD and linear estimates.

We compared the results by analysing the average forecasting errors, the relative forecasting errors and coefficients of determination R^2. These characteristics are well-known measures of the effectiveness in predicting the future outcomes

using regression models and they can be used with parametric and non-parametric models.

We showed first the travel time prediction results (Figure 8) by average forecasting errors using different models (Figure 8). The KD model was c. 5% better than the linear model. There was a strong positive correlation between the linear and KD estimates (about 0.8), but we demonstrated that the 'average' estimate was often slightly better (2%) than the KD estimate.
The 'oracle' estimate is considerably better than the other estimates (c. 40% better than the KD model).

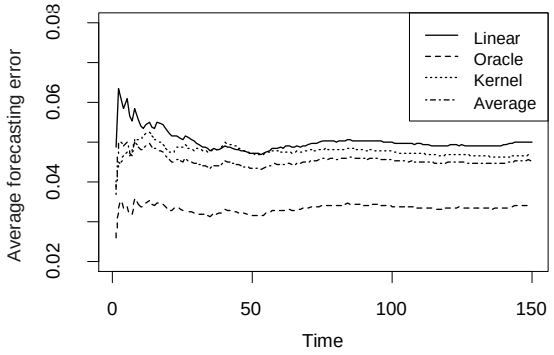


Figure 8 Average forecasting errors using the kernel, linear and combined approaches

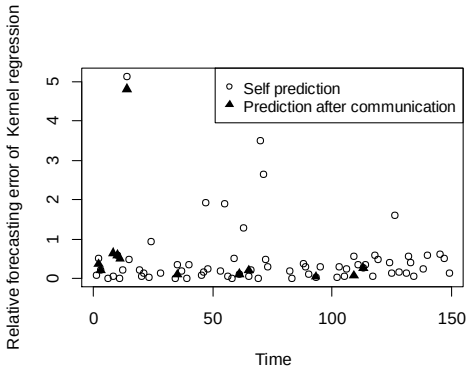


Figure 9 System dynamics of single agent using kernel model

We analysed the system dynamics over time for one randomly selected agent (Figure 9) by considering relative forecasting errors and a number of communication events. Large errors disappeared over time and the number of communication events also decreased.

We compare decentralized coordinated architecture with centralized architecture (data central aggregation) and with decentralized uncoordinated architecture without information exchange between agents. Table 2 summarises the results of the average forecasting errors and goodness-of-fit criterion R^2 values, using various forecasting models (linear, kernel, 'average', and 'oracle').

The linear regression model used all the data points for forecasting. This meant that the straight regression line was being adjusted continuously via coordination and new data points, so it could not be fitted well for all data points. The centralized architecture was worse because a single line could not make good forecasts for a big number of data points. The linear model within the uncoordinated architecture was worse due to the convergence of the parameters of several agents to local optima instead of global optima.

The KD regression model used only neighbouring points for forecasting. Coordination greatly improved the quality of forecasting because new neighbouring data points were provided. The centralised architecture made the best forecasts because there were many nearby points for most requested point. However, absence of experience (uncoordinated architecture) produced bad forecasts. Coordination sufficiently improved the R^2 value.

Analysis of the relative forecasting errors showed that the average estimator in coordinated and uncoordinated architectures for relatively small amounts of data produced relatively better results than KD or linear estimators. The KD model was more accurate on average, but it sometimes yielded highly inaccurate forecasts. The linear model was less accurate, but it did not produce such big outliers. The average method avoided big outliers of the KD model and it provided more accurate forecasts.

Table 2 Average forecasting errors and goodness-of-fit criterion R^2 values for the different forecasting models and architectures

Model	Average forecasting errors	R^2 (After cross-validation)
Linear		
Centralised	0.051	0.829
Uncoordinated	0.054	0.811
Coordinated	0.050	0.819
KD		
Centralised	0.045	0.913
Uncoordinated	0.053	0.814
Coordinated	0.047	0.859
'Average'		
Centralised	0.046	---
Uncoordinated	0.049	---
Coordinated	0.046	---
'Oracle'		
Centralised	0.034	---
Uncoordinated	0.037	---
Coordinated	0.034	---

The 'oracle' estimator provided a possible lower bound for the average forecasting error. This algorithm could improve forecasts by c. 25%. However, it is still unclear, how to choose between KD and linear estimates.

7.2 Decentralised clustering

Second, we considered decentralized clustering of data listed in Table 1. The autonomous KD clustering and the decentralized cooperative clustering algorithm was used in our experiments [54, 55].

We illustrate a data synchronization step. The help-requesting agent asked for help for point A. In Figure 10, one can see how the helping agent clustered the point using its own data, detected the corresponding cluster, and approximated it with the mixture of three normal distributions (shown as ellipses for two-dimensional case with centres in B, C, D). Then the helping agent sent the corresponding parameters to the help-requesting agent. The help-requesting agent added the obtained parameters as data points to its data and made new clustering. This allowed improving clustering similarity of these two agents from 0.011 to 0.037.

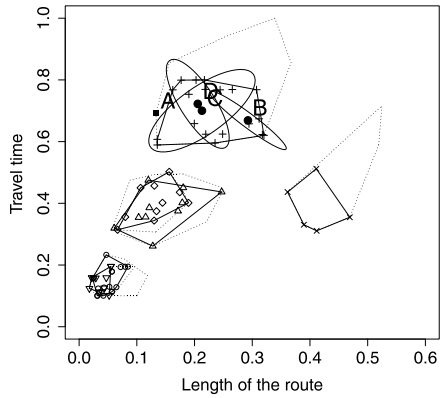


Figure 10 An approximation of a cluster containing A by three ellipsoids with centres in B, C and D

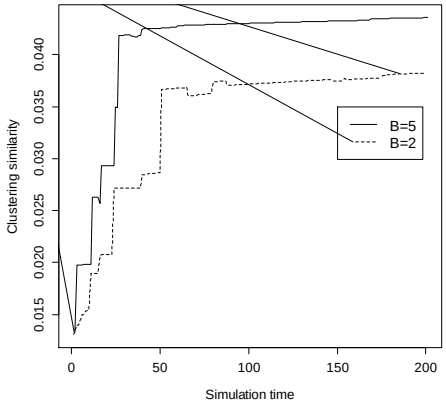


Figure 11 Similarity of agents' clusters over time depending on B components in a mixture of multidimensional normal distributions

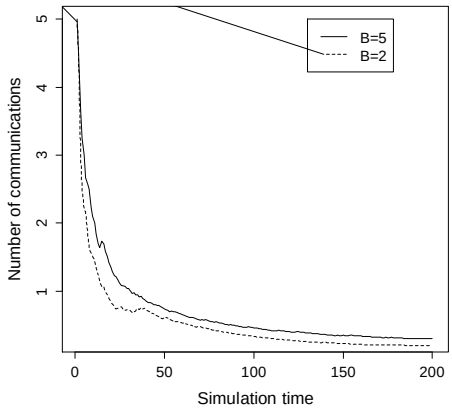


Figure 12 A number of communication events over time depending on B components in a mixture of multidimensional normal distributions

We analysed a system dynamics for a different number of transmitted points. Clustering similarity (Figure 11) increased faster for a bigger number of the estimated and transmitted components of normal distributions B^j for cluster approximations, but the number of communication events (Figure 12) decreased faster.

Note, however, that one communication event was more expensive for a bigger number of transmitted points, but supplied more information.

8.0 FUTURE WORK AND CONCLUSIONS

The main contributions of this study are the following:

- The analysis of related cloud-based architectures and ITS scenarios.
- The definition of the conceptual architecture for implementation of the corresponding BDPM and decision-making strategies for transportation operations.
- The discussion of the employment of appropriate mathematical methods for three ITS scenarios.
- The review of decentralised cooperative computational statistics based BDPM methods and the evaluation of their effectiveness on real-world transportation data.

We envisage this to be an important step towards the next generation of ITS requirements, which were elicited from transportation scenarios. This reflects needs and potential impact of ITS for business and society; the corresponding problems that should be solved for effective cloud system operation were illustrated.

We are planning in our future research to focus on such decentralised BDPM methods as predictive models (regression models of different types, time series, etc.), clustering and classification models, filtering with change point analysis models and BDPM methods for vehicle routing problem.

The near future objectives of our research are:

- *To consider new mathematical models and problems.* Taking into account the success of linear multivariate regression, kernel-based regression, kernel based clustering or resampling tests for change point analysis, resampling travel time estimation for route comparing in traffic routing problem, etc., as referred in the state of the art section, we are planning to implement our decentralised or semi-decentralised approach for other important for the traffic domain and Big data analytics problems (different time series models, logistic regression, multiple regression, etc.).
- *To investigate computational statistics methods for the models described above.* We will use existing and develop new computation intensive methods as a technique for solving the above mentioned problems.
- *To research new agent cooperation techniques.* We are planning to enhance cooperation schemas in the decentralised estimation models involving new reputation computing schemas, new data transmission rules, etc.
- *To implement the described architectures, models, methods and techniques for new ITS applications:* Taking the needs of further transportation scenarios into account, we are planning to apply the developed models, methods and architectures to prediction, clustering, classification and estimation of other stochastic traffic factors.

Acknowledgements

The research leading to these results has received funding from the European Union Seventh Framework Programme (FP7/2007-2013) under grant agreement No. PIEF-GA-2010-274881.

Nomenclature

ICT	- Internet and communication technologies
IoT	- Internet of Things
BDPM	- Big Data processing and mining
ITS	- Intelligent transportation system
FP7	- 7th European Framework Programme for Research and Innovation
EU	- European Union
KD	- Kernel-density
MAS	- Multi-agent system
AmI	- Ambient intelligence
V2V	- Vehicle-to-vehicle
V2I	- vehicle-to-infrastructure
US	- United States of America
ACP	- Artificial, computational, parallel

References

[1] NESSI – Big Data White Paper. December 2012. Big Data - A New World of Opportunities. European Commision.

[2] McKinsey Global Institute. June 2011. Big Data: The next frontier for innovation, competition and productivity. URL: www.mckinsey.com/.

[3] 7-th European Framework Programme project, Instant Mobility. Multimodality for people and goods in urban area. CP 284806. http://instant-mobility.com/.

[4] Gartner. 2013. Gartner Reveals Top Predictions for IT Organisations and Users for 2013 and Beyond. http://www.gartner.com/it/page.jsp?id=2211115.

[5] Talia, D. 2011. Cloud computing and software agents: Towards cloud intelligent services. Proc. of the 12th Workshop on Objects and Agents. 741: 2–6.

[6] Müller, J. P. 1996. The Design of Intelligent Agents. Lecture Notes in Artificial Intelligence. 1177.

[7] Fischer, K., N. Kuhn, and J. Müller. 1994. Distributed, Knowledge-Based, Reactive Scheduling in the Transportation Domain. Proceedings of the Tenth IEEE Conference on Artificial Intelligence for Applications. San Antonio, Texas. 47–53.

[8] Fiosina, J., and M. Fiosins. 2013. Chapter 1: Cooperative Regression-Based Forecasting in Distributed Traffic Networks. In: Memon, Q. A. edited Distributed Network Intelligence, Security and Applications. CRC Press, Taylor and Francis Group. 3–37.

[9] Fiosins, M., J. Müller, and M. Huhn. 2013. A Norm-Based Probabilistic Decision-Making Model for Autonomic Traffic Networks. Highlights on Practical Applications of Agents and Multi-Agent Systems. *Communications in Computer and Information Science.* 365: 49–60.

[10] Foster, I. 2008. Cloud computing and grid computing 360-degree compared. Proc. of the Grid Computing Environments Workshop.

[11] Xiao, L., and Z. Wang. 2011. Internet of Things: a New Application for Intelligent Traffic Monitoring System. Journal of Networks. 6(6):887-894.

[12] Wang, J., J. Cho, S. Lee, and T. Ma. 2011. Real Time Services for Future Cloud Computing Enabled Vehicle Networks. Proc. of the 13th Int. IEEE Annual Conf. on ITS. 1–5.

[13] Li, Z., C. Chen, and K. Wang. 2011. Cloud Computing for Agent-based Urban Transportation Systems. *IEEE Intelligent Systems.* 26(1):73–79.

[14] Wang, F. 2011. Toward a Revolution in Transportation Operations: AI for Complex Systems. *IEEE Intelligent Systems*. 23(6): 8–13.

[15] Fiosins, M., J. Fiosina, and J. Müller. 2012. Change Point Analysis for Intelligent Agents in City Traffic. Agents and Data Mining Interaction. *LNCS.* 7103: 195–210.

[16] Fiosins, M., J. Fiosina, J. P. Müller, and J. Görmer. 2011. Reconciling Strategic and Tactical Decision Making in Agent-Oriented Simulation of Vehicles in Urban Traffic. Proc. of 4th Int. ICST Conf. on Simulation Tools and Techniques (SimuTools'2011). 144–151.

[17] Fiosina, J., M. Fiosins, and J. Müller. 2013. Mining the Traffic Cloud: Data Analysis and Optimization Strategies for Cloud-based Cooperative Mobility Management. Proc. of Int. Sym. on Management Int. Systems. *Adv. in Int. Syst. and Comp*. 220: 25–32.

[18] Fiosina, J., and M. Fiosins. 2012. Cooperative kernel-based forecasting in decentralized multiagent systems for urban traffic networks. Proc. of Ubiquitous Data Mining (UDM) Workshop of ECAI. Montpellier, France. CEUR-WS.org. 960: 3–7.

[19] Fiosina, J. 2012. Decentralised Regression Model for Intelligent Forecasting in Multi-agent Traffic Networks. Proc. of the 9th Int. Conf. on Distributed Computing and Artificial Intelligence. S. Omatu *et al*. (Eds.). *Advances in Intelligent and Soft Computing.* 151: 255–263.

[20] Fiosins, M., J. Fiosina, J. Müller, and J. Görmer. 2011. Agent-based Integrated Decision Making for Autonomous Vehicles in Urban Traffic. *Adv. in Int. and Soft Comp*. 88: 173–178.

[21] Draper, N., and H. Smith. 1998. *Applied Regression Analysis*. Willey.

[22] Box, G., G. Jenkins, and G. Reinsel. 2008. *Time Series Analysis: Forecasting and Control*. Wiley.

[23] Duran, B., and P. Odell. 1974. *Cluster Analysis: A Survey*. London: Springer.

[24] Michie, D., D. Spiegelhalter, and C. Taylor. 2009. *Machine Learning: Neural and Statistical Classification*. Overseas Press.

[25] Chen, J., and A. Gupta. 2011. Parametric Statistical Change Point Analysis: With Applications to Genetics, Medicine, and Finance. Birkhäuser.

[26] Freitas, 2002. *A. Data Mining and Knowledge Discovery with Evolutionary Algorithms*. Springer.

[27] A. L. Symeonidis, and P. A. Mitkas. 2005. *Agent Intelligence Through Data Mining. Multiagent Systems, Artificial Societies, and Simulated Organizations.* New York: Springer-Verlag.

[28] da Silva, J., C. Giannella, R. Bhargava, H. Kargupta, and M. Klusch. 2005. Distributed Data Mining and Agents. *Eng. Appl. of AI.* 18(7): 791–807.

[29] Cao, L., D. Luo, and C. Zhang. 2009. Ubiquitous Intelligence in Agent Mining. Agents and Data Mining Interaction (Lecture Notes in Computer Science). 5680: 23–35.

[30] Klusch, M., S. Lodi, and G. Moro. 2003. Agent-based Distributed Data Mining: The KDEC scheme. *AgentLink.* 104–122.

[31] Zhang, C., Z. Zhang, and L. Cao. 2005. Agents and Data Mining: Mutual Enhancement by Integration. AIS-ADM2005. *LNCS*. 3505: 50–61.

[32] Othmane, B., R. Hebri, and M. Boudiaf. 2012. Cloud Computing and Multi-Agent Systems: A new Promising Approach for Distributed Data Mining. Proc. of the 34th Int. Conf. on Information Technology Interfaces.

[33] Gentle, J. E. 2009. *Computational Statistics*. Springer.

[34] Wu, C. 1986. Jackknife, Bootstrap and Other Resampling Methods in Regression Analysis. *The Annals of Statistics.* 14(3): 1261–1295.

[35] Afanasyeva, H. 2005. Resampling Median Estimators for Linear Regression Model. *Transport and Telecommunication*. 6(1): 90–94.

[36] Afanasyeva, H., and A. Andronov. 2006. On Robustness of Resampling Estimators for Linear Regression Models. *Communications in Dependability and Quality Management: An international Journal*. 9(1): 5–11.

[37] Afanasyeva, H. 2001. Genetics-based Machine Learning Systems for Classification Task. Scientific Proc. of Riga Technical University. 8–16.

[38] Afanasyeva, H. 2002. Fuzzy Learning Classifiers Systems for Classification Task. *Transport and Telecommunication.* 3(3): 43–51.

[39] Afanasyeva, H. 2005. Resampling-approach to a Task of Comparison of Two Renewal Processes. Proc. of 12th Int. Conf. on Analytical and Stochastic Modelling Techniques and Applications, Riga. 13–21.

[40] Andronov, A., and M. Fiosins. 2004. Applications of Resampling Approach to Statistical Problems of Logical Systems. *Acta et Commentationes Universitatis Tartuensis de Mathematica*. 8: 63–72.

[41] Fiosins, M. 2000. Efficiency Of Resampling Estimators Of Sequential-Parallel Systems Reliability. Proc. of the 2nd Int. Conf. on Simulation, Gaming, Training and Business Process Reengineering in Operations. Riga. 112–117.

[42] Bazzan, A., and F. Klügl. 2013. A Review on Agent-based Technology for Traffic and Transportation. *The Knowledge Engineering Review FirstView*. 1–29.

[43] Lin, H., R. Zito, and M. Taylor. 2005. A Review o Travel-time Prediction in Transport and Logistics. Proc. of the Eastern Asia Society for Transportation Studies. 5: 1433–1448.

[44] Malnati, G., C. Barberis, and C. Cuva. 2007. Gossip: Estimating Actual Travelling Time Using Vehicle to Vehicle Communication. 4-th Int. Workshop on Intel. Transportation. Hamburg.

[45] Görmer, J., Ehmke, J., M. Fiosins,. D. Schmidt, H. Schumacher, and Tchouankem, H. 2011. Decision Support for Dynamic City Traffic Management Using Vehicular Communication. Proc. of 1st Int. Conf. on Simulation and Modeling Methodologies, Technologies and Applications (SIMULTECH 2011). 327–332.

[46] Claes, R., and T. Holvoet. 2011. Ad Hoc Link Traversal Time Prediction. Proc. of the 14th Int. IEEE Conf. on Intelligent Transportation Systems. 1803–1808.

[47] Guestrin, C., P. Bodik, R. Thibaux, M. Paskin, and S. Madden. 2004. Distributed regression: an efficient framework for modeling sensor network data. Proc. of the 3rd Int. Sym. on Information processing in sensor networks. New York, USA. 1–10.
[48] Stankovic, S., M. Stankovic, and D. Stipanovic. 2009. Decentralized parameter estimation by consensus based stochastic approximation. *IEEE Trans. Automatic Control*. 56(3): 531–543.
[49] Smith, B., B. Williams, and R. Oswaldl. 2002. Comparison of Parametric and Nonparametric Models for Traffic Flow Forecasting. Transportation Research Part C: Emerging Technologies. 10: 303–321.
[50] Weijermars, W., and E. van Berkum. 2005. Analyzing Highway Flow Patterns Using Cluster Analysis. Proc. of the 8th Int. IEEE Conf. on ITS. Vienna. 831–836.
[51] Lee, J., J. Han, and K. Whang. 2007. Trajectory Clustering: A Partition-and-Group Framework. Proc. of ACM SIGMOD Int. Conf. on Management of Data. Beijing. 593–604.
[52] Hinneburg, A., and H. Gabriel. 2007. DENCLUE 2.0: Fast clustering based on kernel density estimation. Proc. of IDA'07, Adv. in Intelligent Data Analysis VII, LNCS. 4723: 70–80.
[53] Ogston, E., B. Overeinder, M. van Steen, and F. Brazier. 2003. A Method for Decentralized Clustering In Large Multi-agent Systems. Proc. of 2nd Int. Conf. on Autonomous Agents and Multiagent Systems. 789–796.
[54] Fiosina, J., M. Fiosins, and J. Müller. 2013. Decentralised Cooperative Agent-based Clustering in Intelligent Traffic Clouds. Proc. of 11th German Conference on Multiagent System Technologies (MATES 2013), LNCS. (accepted for publication).
[55] Fiosina, J., and M. Fiosins. 2013. Density-Based Clustering in Cloud-Oriented Collaborative Multi-Agent Systems. Proc. of Int. Conf. on Hybrid Artificial Intelligence Systems (HAIS 2013). LNCS. (accepted for publication).
[56] Fiosina, J., and M. Fiosins. 2011. Resampling-based Change Point Estimation. Proc. of the 10th Int. Sym. on Intelligent Data Analysis (IDA'11). *LNCS*. 7014: 150–161.
[57] Fiosina, J., and M. Fiosins. 2013. Selecting The Shortest Itinerary in a Cloud-based Distributed Mobility Network. Proc. of 10th Int. Conf. on Distributed Computing and AI (DCAI 2013). *Adv. in Int. Syst. and Comp*. 217: 103–110.
[58] Fiosins, M. 2013. Stochastic Decentralized Routing of Unsplittable Vehicle Flows Using Constraint Optimization. Distributed Computing and Artificial Intelligence. *Advances in Intelligent Systems and Computing*. 217: 37–44.
[59] Härdle, W., M. Müller, S. Sperlich, and A. Werwatz. 2004. Nonparametric and Semiparametric Models. Berlin/Heidelberg: Springer.
[60] Dempster, A., N. Laird, and D. Rubin. 1977. Maximum Likelihood from Incomplete data via the EM Algorithm. *J. of the Royal Stat. Society. Series B*. 39: 1–38.
[61] Ben-Hur, A., A. Elisseeff, and I. Guyon. 2002. A Stability Based Method for Discovering Structure in Clustered Data. *Pacific Sym. on Biocomputing*. 7: 6–17.
[62] Racine, J. S. 1997. Consistent Significance Testing for Nonparametric Regression. *Journal of Business and Economic Statistics*. 15: 369–379.
[63] Passos, L., R. Rossetti, and E. Oliveira. 2010. Ambient-centred Intelligent Traffic Control and Management. Proc. of the 13th Int. IEEE Annual Conf. on ITS. 224–229.
[64] Kargupta, H., and P. Chan. 2000. *Advances in Distributed and Parallel Knowledge Discovery*. California, USA:AAAI Press/MIT Press.
[65] Lee, W., S. Tseng, and W. Shieh. 2010. Collaborative Real-time Traffic Information Generation and Sharing Framework for the Intelligent Transportation System. *Inf. Sciences*. 180: 62–70.