

Data-driven Framework for Energy-Efficient Smart Cities

Nenad Petrović¹, Đorđe Kocić¹

Abstract: Energy management is one of the greatest challenges in smart cities. Moreover, the presence of autonomous vehicles makes this task even more complex. In this paper, we propose a data-driven smart grid framework which aims to make smart cities energy-efficient focusing on two aspects: energy trading and autonomous vehicle charging. The framework leverages deep learning, linear optimization, semantic technology, domain-specific modelling notation, simulation and elements of relay protection. The evaluation of deep learning module together with code generation time and energy distribution cost reduction performed within the simulation environment also presented in this paper are given. According to the results, the achieved energy distribution cost reduction varies and depends from case to case.

Keywords: Blockchain, Deep learning, Linear programming, Relay protection, Smart grid, Smart city, Smart contract.

1 Introduction

Smart city is defined as incorporation of electronic and digital technologies within the public areas and working environment to transform everyday life with goal to enable sustainable development [1]. However, this transformation brings many challenges related to infrastructure development, building design, transportation, waste management, governance, healthcare, pollution and energy management [1, 2]. In this paper, we focus on energy management in smart city.

The increasing citizens' demand for electrical energy and limited availability of non-renewable resources, on the other side are putting more pressure on the existing energy distribution systems, which are identified as a potential bottleneck for growth of smart cities [3 – 5]. In recent years, a lot of effort has been put in transformation of the existing energy distribution systems driven by information and communication technologies towards the so-called *smart grid*. Smart grid is a two-way cyber-physical system leveraging the collected information by computational intelligence in order to provide clean,

¹Faculty of Electronic Engineering, University of Niš, Aleksandra Medvedeva 14, 18000 Niš, Serbia;
E-mails: nenad.petrovic@elfak.ni.ac.rs; seriusdjoka@gmail.com

safe, secure, reliable, resilient, efficient, economic and sustainable electrical energy to consumers [3 – 5]. For that reason, the adoption of smart grid technologies is of utmost importance for energy efficiency in smart cities [6]. Moreover, the presence of autonomous cars makes the energy management of smart cities even more complex [7].

In this paper, we propose a framework for energy-efficient smart cities leveraging the collected data about the consumers and the environment, focusing on two aspects: energy trading and autonomous vehicle charging. Regarding the implementation, the following elements are presented: linear optimization models for optimal energy trading and vehicle charging relying on blockchain technology, deep learning module for anomaly detection and load forecasting, semantic-driven framework for adaptivity and smart contract generation, the implementation of actuation mechanisms based on relay protection, modeling and simulation tools. When it comes to framework evaluation, the accuracies of anomaly detection and load forecasting of deep learning module are considered, code generation time in context of system response and energy distribution cost reduction.

2 Related Work

The huge potential of data-driven frameworks for energy management in smart cities was shown in several works so far [6, 8]. In [6], the focus is on data-driven decision-making for energy management. On the other side, in [8], the focus is on optimal energy distribution for adaptable consumer demand. However, we cover both aspects and include elements of modeling and simulation for user-friendly evaluation of complex smart city scenarios related to energy management.

Moreover, we propose the adoption of blockchain technology for energy trading. In recent years, the blockchain technology has been adopted in various fields to ensure security and trust, especially when financial transactions are involved. This was also the case for energy trading. However, the existing solutions mainly focus on the trading itself and do not consider the problem of providing energy timely [9 – 11], which is contradictory to the main principles of smart grid architecture advocating that near real-time response is crucial [3 – 5]. Therefore, we decide to focus on development of a mechanism for efficient energy trading between the power generation and the consumers leveraging the load forecasting in synergy with optimization to provide timely response to consumer demands.

3 Background

In this section, the underlying theoretical concepts, approaches and methods used for development of the proposed architecture are presented. For

each of them, a brief overview is given and role in scope of this paper is described.

3.1 Linear optimization

Linear optimization (also known as *linear programming*) is a method aiming to achieve the best possible outcome (maximum profit or lowest cost), based on a mathematical model with requirements represented by linear relationships [12]. More precisely, it refers to a technique for optimization of a linear *objective function*, subject to linear equality and inequality *constraints*. Problem which can be expressed in the canonical form such as the following is referred to as linear program:

$$\text{maximize } c^T x, \quad (1)$$

$$\text{subject to } Ax \leq b, \quad (2)$$

$$\text{and } x \geq 0. \quad (3)$$

In this case, x represents the vector of variables that have to be determined (*decision variable*). Moreover, c and b are vectors of known coefficients, while A is a matrix of known coefficients.

Linear programming is widely adopted – from business and economics to engineering. In engineering, linear programming models are leveraged in transportation, energetics, telecommunications, manufacturing, especially in planning, routing, scheduling, assignment and distribution. In electric grids, optimization is used in processes of production, transmission and consumption to minimize costs, reduce overall consumption, balance supply and demand [13].

In this paper, linear programming is adopted to ensure energy efficiency in smart cities, considering two aspects: energy trading and autonomous vehicle charging. The goal is to minimize the overall energy consumption and related costs. For implementation, we rely on AMPL², an algebraic modeling language for mathematical programming [14]. It provides the ability to write linear programs using the expressions similar to traditional algebraic notation with extensions that provide support for definition of network flow constraints and piecewise linearities specific to optimization problems. However, AMPL itself is not able to solve an optimization problem, but it offers an interface to other programs which act as solvers. In our case, CPLEX³ was used as a solver, whose implementation is based on simplex method [15].

3.2 Deep learning

Deep learning refers to a family of machine learning techniques and algorithms based on artificial neural networks. The learning process itself can be either supervised, semi-supervised or unsupervised. It has been applied in

² <https://ampl.com/>

³ <https://www.ibm.com/analytics/cplex-optimizer>

various areas, such as computer vision, speech recognition, natural language processing, audio recognition, bioinformatics.

Artificial neural network (ANN) is a group of interconnected computational units (referred to as *neurons* or *perceptrons*) through weighted links. Each of them contains one or more weighted input connections, a transfer function that combines the inputs, and an output connection [16]. It receives the signal, processes it and then forwards it to other nodes connected to it. In neural networks, three types of layers are identified [16, 17]: 1) *input layer* – corresponds to the input variables 2) *hidden layers* – the nodes between input and output layers 3) *output layer* - produces the output variables. A *deep neural network* (DNN) is an artificial neural network (ANN) with multiple layers between the input and output layers [17].

In this paper, deep learning approach is used for two aspects: anomaly detection and load forecasting (consumer demand) within the electric grid in smart city. For anomaly detection, the model is trained on historical data of electric measurements. On the other side, for load forecasting, the historical data on power consumption during different periods of year is used together with weather data (temperature, humidity, atmospheric pressure and rainfall). For implementation, we rely on TensorFlow⁴ library for Python programming language provides support for GPU-enabled execution on CUDA-compatible hardware. The load forecasting module builds upon our previous work presented in [18], while the information about daily humidity, atmospheric pressure and rainfall was included beside the average daily temperature as it was shown that inclusion of additional weather data leads to more accurate forecasts [19].

3.3 Blockchain and smart contracts

Since the breakthrough of Bitcoin cryptocurrency in 2009, the underlying blockchain has been considered as one of the most influential technologies of the last decade [20 – 22]. Its main purpose was enabling the transfer of financial assets worldwide without intermediary and additional transfer costs. From music industry to transportation and healthcare, huge efforts are put to adopt blockchain within the existing information systems and applications in order to increase overall safety and trust.

Blockchain itself refers to a data structure (also called *ledger*) that consists of append-only sequence of blocks holding the information about the executed transactions [21]. However, the same term is often used for a distributed system that stores copies of the previously mentioned data structure within the peer-to-peer network of nodes. Each node has alphanumeric address that enables both anonymity and transaction record transparency at the same time. Moreover,

⁴ <https://www.tensorflow.org>

each block also contains a cryptographic hash of the previous block and timestamp in order to ensure that no one can modify or delete them, once they are recorded in ledger. The transaction itself represents transfer of value and ownership of digital tokens between sender and recipient recorded in the distributed ledger [21, 22]. Token can represent either tangible or intangible goods/assets, from cash and electrical energy to intellectual property.

When it comes to blockchain's adoption within the smart grid systems, the use cases presented in current literature can be summarized as follows [23, 24]: 1) operational status log of power grid 2) decentralized control of power transmission and distribution 3) energy trading. In this paper, we focus on blockchain adoption within the processes of energy trading and autonomous car charging.

Smart contracts are recognized as a key-enabler of these use cases [24]. Smart contract is a protocol intended to digitally facilitate, verify, or enforce the execution of a particular contract [22]. In context of blockchain technology, it refers to a software code that defines and executes transactions on the target platform, while the performed transactions are trackable, irreversible and do not involve third parties. It can be implemented in any programming language and secured using encryption and digital signature. In case of Ethereum [25] blockchain (considered in this paper), smart contracts are written using a high-level object-oriented language Solidity⁵, developed by Ethereum Foundation. We decide to use Solidity due to fact that it is far more expressive and powerful than Bitcoin's original script language used for definition of smart contracts.

A similar approach to energy trading leveraging Solidity contracts within Ethereum blockchain network was used in [26]. The smart contract generation module used in this paper builds upon the semantic-driven Solidity smart contract generator for Ethereum platform presented in [27] which originally aimed contracts between the participants in music industry supply chain.

3.4 Ontologies and semantic technology

The term *ontology* has wide use in different scientific fields. Initially, it referred to a philosophical branch studying the ways of being, basic concepts of being and their relations. In computer science, ontology is a vocabulary of terms used to represent certain knowledge. This implies formalization of knowledge and representation in a form suitable for use by both humans and computers. In this context, ontology is often defined as a representational artifact, comprising a taxonomy as proper part, whose representations are intended to designate some combination of universals, defined classes, and relations between them [28]. Every ontology consists of *classes*, *individuals*, *attributes* and *relations*. Classes are abstract groups, collections, types of objects, while individuals are

⁵ <https://solidity.readthedocs.io/en/v0.5.5/>

instances of these classes. Attributes are related properties, characteristics and parameters of classes. Relations define ways in which classes and individuals can be related. Collection of facts specified according to the conceptualization defined by some ontology is often stored separately from the corresponding ontology and referred to as *knowledge base*. Ontology is augmented with a set of *rules* that are used to generate new knowledge from the existing set of facts.

The role of semantic technology is to encode the meaning of data separately from its content and application code. This way, it is enabled for machines to understand data, exchange the understanding and perform reasoning on top of it. In the context of semantic technologies, ontologies are used to describe the shared conceptualization of a particular domain [29]. Semantic descriptions are represented using the RDF⁶ standard language in the form of triplets (*subject, predicate, object*) and persisted inside triple stores. SPARQL⁷ is a language used for querying the RDF semantic triple stores. By executing queries against the triple store, the retrieved results can be used in order to support different reasoning mechanisms for inference of new knowledge based on the existing facts.

4 Architecture Overview

First, the measured electrical signals and other measurements are acquired and stored into the measurements database for a certain period. The measurement of electrical and other useful quantities is performed by various IoT and smart devices, often referred to as *smart meters*. Among them, we have low-power single-board computers (such as Raspberry Pi) or smartphones [30].

After that, the deep learning module performs training based on the historical data, so it becomes able to make predictions about the consumer demand within the electric grid in future and detect anomalies. The obtained results are semantically annotated, so the semantic reasoning can be leveraged to draw conclusions about the events that occurred within the smart grid environment. According to these results, the corresponding actions are taken in order to adapt the smart grid to current consumption demands and respond these events.

A set of actions that has to be taken when specified condition is fulfilled (referred to as *adaptation plan*) is specified by operators, using a visual modeling tool relying on a domain-specific notation. Finally, the device-specific commands are generated in order to respond to the changes detected in smart grid.

On the other side, the load forecast produced by the deep learning module is forwarded to linear optimization component, that is taking them as input,

⁶ <https://www.w3.org/RDF/>

⁷ <https://www.w3.org/TR/rdf-sparql-query/>

together with the other information in order to calculate the optimal energy distribution. The energy trading matrix is produced as output of the optimization process, holding the information about the amount of energy that needs to be transferred between the grids. Finally, this matrix is used for generation of smart contract code that enables the execution of blockchain-based transactions. The proposed architecture is illustrated in Fig. 1.

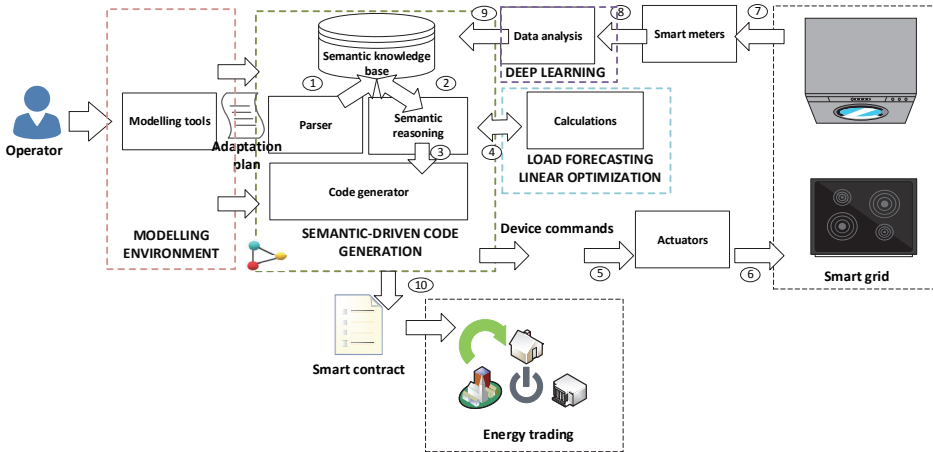


Fig. 1 – Overview of the data-driven architecture for energy-efficient Smart Cities. 1: Semantic annotations of adaptation strategy, 2: SPARQL queries and query results, 3: Command parameters, 4: Input data for calculation and its results, 5: Device-specific command script, 6: Voltage/current regulation and turning the devices on/off, 7: Events and messages, 8: Electric and other measurements, 9: Semantic annotations of data analysis results 10: Transaction execution in targeted blockchain.

5 Linear Optimization Models

5.1 Efficient energy trading

A possible approach to energy efficiency in smart cities is energy trading. It enables customer participation in decentralized bilateral energy trading to fulfill their needs without additional stressing of the power grid. There are many benefits of efficient energy trading [31, 32]: 1) the improvement of the overall system efficiency 2) reduction of operational costs 3) emissions reduction by leveraging the renewable sources as much as possible. For that reason, we propose a linear optimization model for efficient energy trading between the power generators – both the power grids and prosumers (such as households that both produce and consume energy).

Let us consider a network of interconnected energy producers/consumers denoted as N , where each of them is referred to as *node*, while the links between nodes are denoted as *arcs* in graph. Each node n_i is characterized by the

maximum capacity of electrical energy that can be produced, denoted as c_i and consumer demand for electricity within that grid, denoted as d_i . Both of them are given in megawatts. In cases when consumer demand exceeds the available power that can be produced by the considered node, the process of energy trading is performed. To each arc, a decision variable indicating the amount of energy that will be exchanged between the nodes is assigned:

$$\begin{aligned} x[i, j] &\geq 0 \text{ if energy trading between } n_i \text{ and } n_j \text{ will be performed,} \\ x[i, j] &= 0 \text{ otherwise.} \end{aligned} \quad (4)$$

For each arc, the cost of energy transfer between n_i and n_j is referred to as $tc[i, j]$ given in dollars per megawatt. Furthermore, the power generation cost of each n_j is denoted as $gc[j]$, given in the same unit as transfer cost.

The main constraint related to energy trade between nodes is that the amount of overall energy available is equal or greater than the demand for each node within the network. The overall energy of each node is a sum of its capacity, positive value of energy received from other nodes and negative value of energy sent to other nodes. This constraint can be expressed as:

$$c[i] + \sum_{j \in N} x[i, j] - x[j, i] \geq d[i], \quad i \in N. \quad (5)$$

When it comes to objective function, a possible approach is to minimize the sum of costs, taking into account both the power generation and energy trading between the nodes to optimize the energy distribution:

$$\text{minimize } \sum_{i, j \in N} x[i, j] tc[i, j] gc[i]. \quad (6)$$

5.2 Autonomous vehicle charging

Let us assume that there is a set S of charging stations that is connected to a common electric grid. Each station s_j is characterized by the maximum power that it can provide in order to charge the battery of an autonomous car, referred to as mp_j . Moreover, to each s_j we associate the cost of electric energy distribution, denoted as dc_j .

On the other side, there is a set of autonomous vehicles, denoted as V . To each vehicle v_i , we associate the maximum battery charge (mc_i), current battery charge (cc_i) and energy spending per km (tc_i). Moreover, the length of paths between each v_i and s_j are denoted as $d[i, j]$. For each arc, we assign a decision variable $x[i, j]$:

$$x[i, j] = 1 \text{ if } v_i \text{ will recharge at } s_j, \quad x[i, j] = 0 \text{ otherwise.} \quad (7)$$

When it comes to objective function, the goal to minimize the overall sum of costs, taking into account both the distribution cost and travel cost for a given distance, given as:

$$\text{minimize } \sum_{i \in V, j \in S} x[i, j](dc[j] + tc[i]d[i, j]). \quad (8)$$

However, there are several constraints. First, the vehicle's current charge must be enough to arrive to the charging station:

$$\sum_{j \in S} x[i, j]tc[i]d[i, j] \leq cc[i], \quad i \in V. \quad (9)$$

Moreover, each station can accept one vehicle at a time:

$$\sum_{i \in V} x[i, j] \leq 1, \quad j \in S. \quad (10)$$

and each vehicle can be charged at one station at a time:

$$\sum_{j \in S} x[i, j] = 1, \quad i \in V. \quad (11)$$

Finally, the station must provide enough energy to the vehicle that is being charged:

$$mc[i]x[i, j] \leq mp[j], \quad \text{where } i \in V \text{ and } j \in S. \quad (12)$$

6 Deep Learning Module

6.1 Load forecasting

The input data is split into two disjoint datasets – training (80% of data) and test (20% of data). It consists of five variables, as shown in **Table 1**.

Table 1
Load forecasting dataset.

Consumer	Day	Rainfall [mm/m ²]	Humidity [%]	Pressure [Pa]	Temp [°C]	Consumption [MW]

Consumer is the identification number of power consumer. Day refers to the ordinal number of a day within the considered year. Rainfall refers to the amount of liquid precipitation over an area in that day. Humidity refers to the concentration of water vapour present in air expressed in percentage. Furthermore, we include the pressure within the atmosphere of Earth. Temperature refers to the average value of temperature recorded the considered day. Finally, the consumption is the amount of energy spent during the considered day by the given consumer. Consumer, day, rainfall, humidity,

pressure and temperature are considered as independent variables, while the energy consumption is treated as dependent variable (that is being predicted).

For evaluation, a public domain dataset about the household consumption was used [33]. The data was prepared by summing the value of consumption by all the households within the grid for each day. After that, the information about the weather that day in the given geographic location of the grid was added to the dataset from OpenWeather [34].

6.2 Anomaly detection

The electric measurement data used for anomaly detection was acquired from consumers (mainly from typical household devices, such as heater, washing machine and fridge) using the system based on Android smartphones presented in [30]. Voltage and frequency of electric signal coming from consumers were measured. In the training dataset, they are treated as independent variables. On the other side, the label denoting whether anomaly of electric signal is detected was added to the dataset (dependent variable). In the training dataset, the label “0” stands for the measurements during normal operation of consumer devices, while “1” stands in case when there is anomaly. The labels for the measurements were added manually to the training set.

The whole set of measurements acquired from consumers was divided into training (80%) and test dataset (20%). The task of deep learning module is to determine whether the new measurement contains an anomaly or no leveraging the knowledge learned from previous observations from the training set. The structure of dataset used for anomaly detection is shown in **Table 2**.

Table 2
Anomaly detection dataset.

Consumer	Voltage [V]	Frequency [Hz]	Anomaly [0/1]
----------	----------------	-------------------	------------------

7 Semantic Framework

7.1 Smart Grid Response Ontology

A domain ontology (illustrated in Fig. 2) is defined with purpose to semantically annotate the results obtained as output of data analysis module. This way, we provide means to enable reasoning about the changes (events) that occurred by executing SPARQL queries and interpreting their results. Various types of events are considered, such as voltage anomaly and device failure. Furthermore, for each of the available events, there are possible actions to be taken in order to adapt to detected events, such as voltage regulation, turning off the device, switching the device to power saving mode. Each device is

monitored by corresponding smart meter and regulated by actuator device. The actuator-specific commands are generated using this ontology and applied to target consumer devices by corresponding actuators which regulate them.

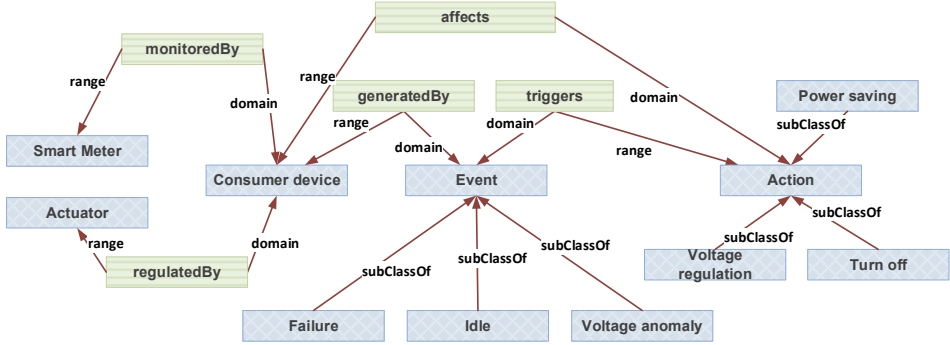


Fig. 2 – Smart Grid Response Ontology.

According to this ontology, each command generation rule r_i is described by $(event_e, action_a, device_d, actuator_{ac})$. The device-specific commands are generated as a response to the events that occur, as given by formula:

$$r_i : \text{if}(event_e) \text{ then generate}(action_a) \text{ for actuator}_{ac} \text{ regulating device}_d . \quad (13)$$

7.2 Energy Trading Ontology

Each smart contract for energy trading consists of two types of participants – buyer and generator. Buyer refers to a participant that will receive an amount of energy by paying the overall price of the trade. Generator is the participant that sends the given amount of energy, but, on the other side, receives the amount of tokens equivalent to the trade price of energy transferred. The trade price is defined as:

$$\frac{(a \cdot tc \cdot gc)}{tp} . \quad (14)$$

In this case a is amount of energy that will be traded, tc is transfer cost, gc is generation cost and tp is token price. While transfer cost and generation cost are characteristics of the energy generator, token price refers to the current price of a cryptocurrency token in dollars. Moreover, the token balance and available energy of both the buyer and the generator before and after the transaction execution are captured for purpose of smart contract verification.

Once the semantic representation of energy trading transaction is constructed (according to the ontology shown in Fig. 3), the corresponding smart contract is generated by executing SPARQL queries which return results that are used to populate the parameters of a template.

The main advantage of using semantic representation of energy trading smart contracts is to enable interoperability and integration of heterogeneous blockchain and smart grid platforms by using appropriate code generators.

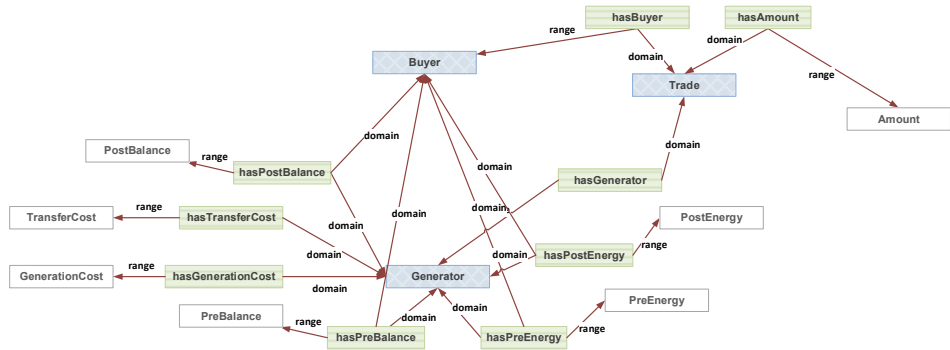


Fig. 3 – Energy Trading Ontology.

8 Adaptation plan Modelling Tool

A visual modelling tool providing the ability to define the adaptive behavior of smart grid using a domain-specific notation was developed within the Node-RED⁸ environment. The targeted users are smart grid operators.

Domain-specific languages and notations are specialized for solving problems from a particular application domain. Metamodel is a model of a modelling language defining the structure and constraints for a family of models. The domain-specific notation within the tool is described by a metamodel shown in Fig 4.

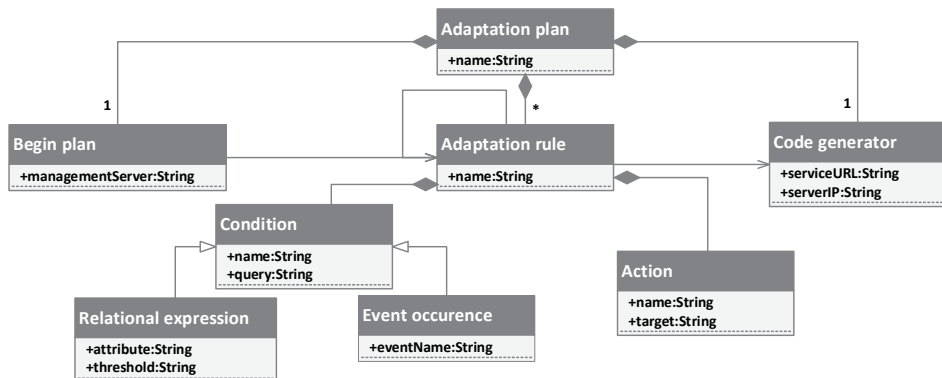


Fig. 4 – Adaptation plan metamodel as UML class diagram.

⁸ <https://nodered.org>

In this paper, the proposed notation is used to define a set of actions that will be taken over the targeted devices in order to adapt smart grid to the environment changes when pre-defined environment conditions are satisfied. The conditions could be either some specific events or relational expressions with respect to a given threshold.

Node-RED is an intuitive and extendable framework that is used for wiring together IoT devices, APIs and online services in novel ways, providing a browser-based editor with drag-and-drop user interface using the wide range of modeling elements (nodes). Therefore, in order to enable adaptation strategy definition within Node-RED, we had to develop custom node elements to embed the described domain-specific notation.

Three types of custom nodes are developed and added to Node-RED's modeling environment: *begin_plan*, *adaptation_rule* and *code_generator*. Begin plan is the element that opens a new adaptation plan and includes the choice of the server responsible for management of the targeted smart home or smart grid. Adaptation rule defines (*condition*->*action*) mapping expressing the fact that when condition is fulfilled, the corresponding action will be triggered to change the state of a target device. Code generator is an element used for closing of adaptation plan and provides parameters needed to invoke the code generation process deployed on edge server. This way, the modelling tool remains independent from the implementation of backend and command generation mechanisms. In Fig. 5, an example of adaptation plan is given in our modelling environment based on Node-RED. The shown rule turns off the targeted heater in case of voltage anomaly.

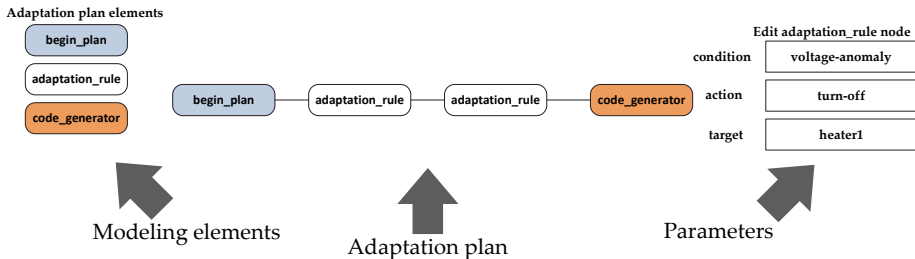


Fig. 5 – Adaptation plan modelling tool.

9 Code Generation Algorithms

9.1 Response command generation

The role of response command generation algorithm is to construct the device-specific commands that are used to respond to the changes that occur by the actuator devices.

First, the measured data is analysed applying the available data analysis techniques. In our case, the detection of anomalies is performed by deep learning module. After that, the obtained results are semantically annotated according to the Smart Grid Response Ontology (shown in Fig. 2). In semantic reasoning phase, each condition from the adaptation plan is translated to a SPARQL query which is executed against the semantic knowledge base. If the query results are interpreted in a way that condition is fulfilled, then the corresponding device-specific command is generated in order implement the selected action for the target device and appended to the output command script. The pseudo-code of the described algorithm is shown in Listing 1.

Input: measured data, adaptation plan
Output: device-specific command script
Steps:

1. Analyse measured data;
2. Semantically annotate the results obtained in Step 1 using the Smart Grid Response Ontology;
3. Retrieve all the adaptation rules from the adaptation plan;
4. For each adaptation rule in adaptation plan
5. query_results=executeQuery(adaptation_rule.condition.query);
6. condition_truth=interpret(query_results);
7. if(condition_truth is true) then
8. Generate command for adaptation_rule.action targeting the actuator of adaptation_rule.target;
9. Append generated command to the output script;
10. end if
11. end for each
12. end

Listing 1 – *Response command generation algorithm pseudo-code.*

9.2 Smart contract generation for energy trading

The role of smart contract generation algorithm is to parametrize the targeted blockchain technology smart contract templates for energy trading. These contracts enable the execution of transactions according to the optimal energy distribution plan, obtained as output of optimization processes described by models presented in subsections 5.1 and 5.2. In what follows, the smart contract generation procedure shown in Listing 2 will be described.

Input: AMPL .DAT file, AMPL .MOD file optimization model for energy trading
Output: smart contracts for energy trading
Steps:

1. Perform optimization using CPLEX solver for given .DAT and .MOD files;
2. Write energy trading matrix obtained as a result of optimization;
3. Semantically annotate the obtained optimization results;
4. Retrieve all energy trades from the semantic knowledge base;
5. For each trade in energy trades
6. Retrieve trade parameters – buyer, generator, amount, transfer_cost, generation_cost;
7. Insert trade parameters to a smart contract template for target blockchain platform;
8. Generate smart contract;
9. end for each
10. end

Listing 2 – *Smart contract generation algorithm pseudo-code.*

First, the optimization problem for a given scenario written in .DAT file is solved for the corresponding model using AMPL's interface to CPLEX solver. The energy trading matrix is created as a result of optimization process. After that, the obtained matrix is semantically annotated according to the Energy Trading Ontology (shown in Fig. 3) and inserted to the semantic knowledge base. Furthermore, the energy trades are retrieved from the semantic knowledge base. For each trade, the necessary parameters are collected. In our case, these parameters are: buyer and seller address, amount of energy that will be transferred, cost of energy transfer and cost of energy generation. Finally, these parameters are inserted to the targeted blockchain platform smart contract template. In this paper, we use Solidity smart contract template for Ethereum blockchain platform, as shown in Listing 3.

```
pragma solidity ^0.4.21;
contract EnergyTrade{
    event Sent(address buyer, address generator, uint amount, uint
transfer_cost, uint generation_cost);
    uint price;
    uint token_price;
    function trade() public {
        price=(amount*transfer_cost*generation_cost)/token_price;
        if (balances[buyer] < price) return;
        balances[buyer] -= price;
        balances[generator] += price;
        emit Sent(buyer, generator, amount, transfer_cost, generation_cost);
    }
}
```

Listing 3 – Solidity smart contract template for energy trading.

10 Actuation

The actuation mechanism used in this paper is provided by relay protection system based on Arduino Uno platform shown in Fig. 6.

Relay protection is the key for safe operation of electric grids [35]. The goal is to detect and localize the anomaly within the electric grid and detach the faulty devices timely [36].

Arduino Uno⁹ is an affordable 8-bit microcontroller, with 16MHz clock, containing 32KB FLASH, 2KB SRAM, 1KB EEPROM, 16 digital in/out pins, 6 analog inputs and equipped with USB connector. The device is powered by 5V. For relay protection, we use an affordable single-channel relay TONGLING JQC-3FF-S-Z¹⁰. For Wi-Fi communication, ESP8266 microchip that supports full TCP/IP stack was used. The system receives MQTT¹¹ messages about the detected anomalies and executes the corresponding commands to detach the

⁹ <https://store.arduino.cc/arduino-uno-rev3>

¹⁰ <https://ru.aliexpress.com/item/32480128984.html>

¹¹ <http://mqtt.org>

faulty consumer devices from the smart grid. The potential of Arduino Uno in such system was approved in [37].

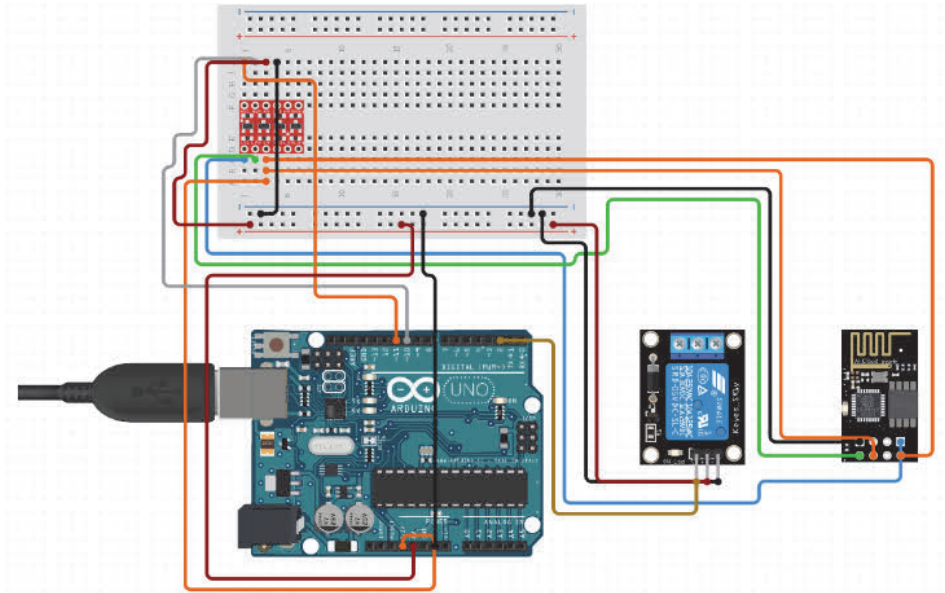


Fig. 6 – Relay protection system based on Arduino Uno.

11 Simulation Tool

The complexity of smart grid and smart city architectures brings many challenges and issues, especially when it comes to reliability, resilience, integration and security of the whole infrastructure. Therefore, the approach of software modelling and simulation can be highly beneficial in process of testing of such applications and mechanisms before the actual deployment to ensure proper operation in all scenarios without high costs [38, 39].

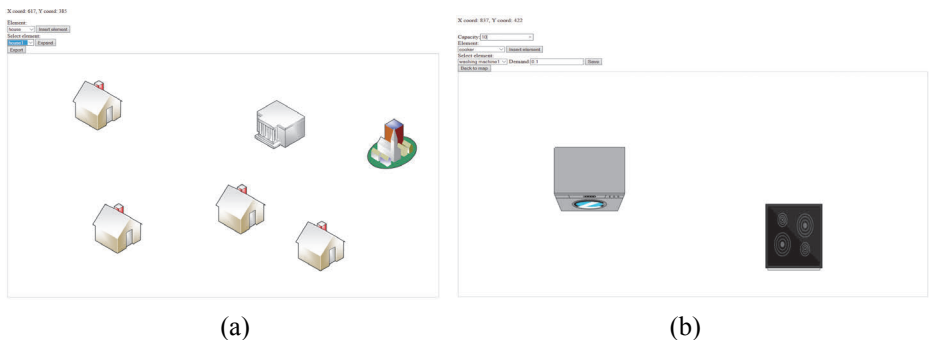


Fig. 7 – Smart grid and smart city simulation tool: (a) map view; (b) internal view.

For that reason, we provide a web browser-based tool implemented mainly using JavaScript, jQuery and HTML that provides the ability to model smart city for scenario testing and evaluation purposes. The screenshots of the proposed simulation tool are shown in Fig. 7.

As it can be seen, there are two different views. The first view (a) provides the ability to place prosumers (that both consume and produce energy), such as households and buildings on grid map. For each of them, there is another view (b) that gives the ability to insert the more detailed information about their power generation capacity, power generation cost, consumer devices and demand for each of the consumer devices. Moreover, the transfer cost between the prosumers can be defined by placing arcs between them. Once the modelling is finished, the AMPL data file that will be used as an input of optimization process can be generated.

A similar solution for smart grid simulation was presented in [39]. However, we also include the modelling elements for scenarios involving the charging of autonomous vehicles. For each autonomous vehicle, it is possible to set the parameters about maximum charging capacity, current battery charge, travel cost per kilometer and speed is included. On the other side, for charging stations, the information about distribution cost and maximum power has to be set.

Furthermore, the mechanisms of dynamic adaptivity based on adaptation plan built upon [40] are integrated. The events that occur within the simulation environment trigger the adaptivity mechanisms that perform code generation as a response. These events can be triggered either manually (for fixed-position consumers) or automatically during the position change of movable elements by using keyboard arrows. The response is then applied to the simulation environment, so the effects of code generation become visible to the user.

12 Evaluation

In this section, the performance evaluation of proposed data-driven framework for energy-efficiency in smart cities is presented. First, the evaluation of deep learning module is given for both anomaly detection and load forecasting. Moreover, the time needed to detect the changes within the smart city environment and respond by generating the corresponding code is considered. Finally, the achieved energy distribution cost reduction is presented.

The evaluation was performed on a laptop equipped with Intel i7 7700HQ CPU, NVIDIA GTX1050 GPU with 2GB VRAM and 16GB of DDR4 RAM. The backend of the platform for change detection and code generation was written in Java, while the deep learning module was developed in Python and executed leveraging GPU. The tuning of deep neural network parameters was performed using similar observations as in [41].

12.1 Load forecasting

The deep learning-based load forecasting module is evaluated with respect to mean relative error. For evaluation purposes, a public domain dataset about the household consumption was used [33] and weather from [34]. In TensorFlow, a neural network with 4 hidden layers was created. The learning was performed in 10 epochs.

As average of 100 runs, the results presented in **Table 3** were achieved. According to these results, we can see that the mean relative error of predicted value decreases as the training set size becomes larger. Moreover, it was noticed that the prediction performance is improved (mean relative error is decreased for around 10%) with respect to our previous work [18], as we also included the rainfall, humidity and atmospheric pressure data within the training set.

Table 3
Load forecasting evaluation.

Training set size	Test set size	MRE [%]	Execution time
80	20	8.96	0.42
800	200	6.01	0.59

12.2 Anomaly detection

The mechanism for detection of electric signal anomalies based on deep learning is evaluated considering the percentage of correctly classified measurements. The measurements were collected using the solution presented in [30]. In TensorFlow, a neural network with 2 hidden layers was created, while the learning process was performed in 10 epochs.

The achieved results are given in **Table 4**. According to them, it can be noticed that the percentage of correctly classified measurements increases as the training set size becomes larger.

Table 4
Anomaly detection evaluation.

Training set size	Test set size	Correct/Test size [%]	Execution time
80	20	90.13	0.31
800	200	96.02	0.39

12.3 System response and energy distribution cost reduction

The proposed solution was evaluated in three scenarios: relay protection, energy trading and autonomous vehicle charging. The achieved results are given in **Table 5**.

The first column shows the scenario name. The next two columns are used for representation of adaptation rules used within the considered scenario. The second column shows the condition to detect the environment change, while the third column is the action that has to be taken as a response to that change. The fourth column is the number of consumers involved into scenario. Moreover, the fifth column shows the time needed for detection of the environment changes by executing corresponding SPARQL queries according to the condition specified within the second column. The sixth column represents the time needed for code generation of the response action previously specified in the third column. Finally, the last column shows the energy distribution cost reduction if applicable for the considered scenario (in case of energy trading and autonomous vehicle charging).

Table 5
System response and energy distribution cost reduction.

Scenario	Condition	Action	Consumers	Detection [s]	Code generation [s]	Cost reduction [%]
Relay protection	voltage-anomaly	turn-off	1	0.42	1.01	-
			2	0.62	1.41	-
			3	0.73	1.04	-
				0.62	1.01	-
	demand-insufficient	turn-off	2	0.69	1.03	-
Energy trading	demand-insufficient	optimal-trading	3	0.51	5.18	79
			5	0.69	6.91	62
			8	0.75	5.12	91
Vehicle charging	battery-low	optimal-charging	3	0.53	8.25	102
			5	0.82	7.11	44
			8	0.94	9.38	59

Regarding the results in **Table 5**, it can be noticed that the code generation is much shorter in relay protection scenario. This can be explained by the fact that the generation of Arduino Uno procedure arguments requires much less computation time than energy trading and autonomous vehicle charging that involve the steps of linear optimization and smart contract generation. On the other side, it is noticeable that the largest code generation time was recorded for vehicle charging scenarios. This can be explained by the fact that this scenario might include two-step optimization – both for battery charging and energy trading.

Moreover, the time necessary for anomaly detection increases with the number of consumers involved as the amount of data which has to be analyzed also grows. However, the same does not always hold for code generation time. It can be noticed that larger number of devices involved does not necessarily lead to larger amount of commands to be generated, as commands are generated only for the devices which satisfy the condition given by the adaptation rule.

Finally, the time needed for detection of environment changes increases with the number of adaptation rules, as it can be noticed in the fourth experiment for relay protection scenario. It can be explained by the fact that each new adaptation rules introduces additional SPARQL query to be executed which requires more time.

13 Conclusion and Future Work

In this paper, a data-driven framework leveraging deep learning and linear optimization for blockchain-supported energy trading was presented. According to the results, energy distribution cost was reduced in all of the relevant scenarios. However, the percentage of reduction depends on the concrete instance of the considered problem. The same is for the code generation time.

Despite the promising results of the work presented in this paper, there are still many important aspects to be considered. First, the issues related to security in smart grids are not covered. However, security is of utmost importance in IoT-based smart grid architecture, as device communication is often performed using vulnerable wireless channels [42]. In this case, the attacks could have catastrophic consequences, such as nation-wide power outage and physical damage to infrastructure and beings. Moreover, the verification of generated smart contracts [43, 44] for energy trading is also a possible research direction in near future.

14 Acknowledgements

This work was partially supported by grants III44006 and III47003 by the Ministry of Education, Science and Technological Development of the Republic of Serbia. Moreover, this paper is an extended version of the work published

under the name “Podacima-vođena arhitektura za prilagodljive energetske mreže zasnovana na IoT uređajima” at the 63rd Meeting of the Society for Electronics, Telecommunications, Computers, Automatic Control and Nuclear Engineering ETRAN (Silver Lake, Serbia, June 3–6, 2019), and which was awarded as The Best Young Researcher’s Paper at the Computing and Information Engineering section.

15 References

- [1] M. Stone, J. Knapper, G. Evans, E. Aravopoulou: Information Management in the Smart City, *The Bottom Line*, Vol. 31, No. 3/4, September 2018, pp. 234 – 249.
- [2] M. Hamalainen: A Framework for a Smart City Design: Digital Transformation in the Helsinki Smart City, *Entrepreneurship and the Community: A Multidisciplinary Perspective on Creativity, Social Challenges, and Business*, August 2019, pp. 63 – 86.
- [3] H. Gharavi, R. Ghafurian: Smart Grid: The Electric Energy System of the Future, *Proceedings of the IEEE*, Vol. 99, No. 6, June 2011, pp. 917 – 921.
- [4] A. H. Bagdadee, L. Zhang: Smart Grid: A Brief Assessment of the Smart Grid Technologies for Modern Power System, *Journal of Engineering Technology*, Vol. 8, No. 1, January 2019, pp. 122 – 142.
- [5] E. Kabalci, Y. Kabalci: *Smart Grids and Their Communication Systems*, Springer, Singapore, Singapore, 2018.
- [6] J. J. Garcia, J. J. Cardenas, R. Enrich, D. Ilić, S. Karnouskos, R. Sauter: Smart City Energy Management via Monitoring of Key Performance Indicators, *Proceedings of the CIRED Workshop 2014*, Rome, Italy, Paper 0263, June 2014, pp. 1 – 5.
- [7] A. Lasisi, F. Ali, N. Attoh-Okine: A Case Study of Multi-Dimensional Emission Data in the Light of Electric Vehicles in Smart Cities: Energy and Policy Implications, *Journal of Sustainable Cities and Society*, May 2019, pp. 1 – 33.
- [8] G. Zheng, B. Zhao, X. Zhao, H. Li, X. Huo, W. Li, Y. Xia: Smart City Energy Interconnection Technology Framework Preliminary Research, *Proceedings of the IOP Conference Series: Earth and Environmental Science*, Vol. 108, January 2018, Paper 052043, pp. 1 – 7.
- [9] C.- G. Lim, S.- T. Lim, I. Aliyu: Blockchain for Small Scale Energy Trading, *Proceedings of the Fall Conference of Electronics and Telecommunications Society in 2018*, Seoul, South Korea, 2018, pp. 59 – 61.
- [10] X. Dong, W. Zaoyu, M. Hua, X. Jing, Y. Debo, W. Fanjin, B. Wei: ETSB: Energy Trading System Based on Blockchain, *Proceedings of the International Conference on Security with Intelligent Computing and Big-data Services*, Vol. 895, September 2019, pp. 673 – 684.
- [11] S.- C. Oh, M.- S. Kim, Y. Park, G.- T. Roh, C.- W. Lee: Implementation of Blockchain-Based Energy Trading System, *Asia Pacific Journal of Innovation and Entrepreneurship*, Vol. 11, No. 3, December 2017, pp. 322 – 334.
- [12] G. B. Dantzig, M. N. Thapa: *Linear Programming 1 – Introduction*, Springer, New York, USA, 1997.
- [13] R. Fourer, D. M. Gay, B. W. Kernighan: *AMPL: A Modeling Language for Mathematical Programming*, 2nd edition, Duxbury Press, USA, 2003.

- [14] M. Ahat, S. B. Amor, M. Bui, A. Bui, G. Guérard, C. Petermann: Smart Grid and Optimization, *American Journal of Operations Research*, Vol. 3, No. 1, January 2013, pp. 196 – 206.
- [15] G. B. Dantzig, A. Orden, P. Wolfe: The Generalized Simplex Method for Minimizing a Linear Form Under Linear Inequality Restraints, *Pacific Journal of Mathematics*, Vol. 5, No.2, 1955, pp. 183 – 195.
- [16] R. Lippmann: An Introduction to Computing with Neural Nets, *IEEE ASSP Magazine*, Vol.4, No. 2, April 1987, pp. 4 – 22.
- [17] Y. Bengio: Learning Deep Architectures for AI, *Foundations and Trends in Machine Learning*, Vol. 2, No. 1, November 2009, pp. 1 – 127.
- [18] N. Petrović, Đ. Kocić: Data-Driven Architecture for Adaptable Smart Grid Based on IoT Devices, *Proceedings of the 63rd National Conference on Electrical, Electronic and Computing Engineering - ETRAN 2019*, Silver Lake, Serbia, June 2019, pp. 880 – 885. (in Serbian).
- [19] A. B. M. Shawkat Ali, S. Azad: Demand Forecasting in Smart Grid, *Green Energy and Technology*, Demand Forecasting in Smart Grid, Green Energy and Technology, Vol. 132, July 2013, pp. 135 – 150.
- [20] A. Ushmani: Blockchain Insight, *International Journal of Computer Science Trends and Technology*, Vol. 7, No. 2, March - April 2019, pp. 1 – 3.
- [21] A. Narayanan, J. Clark: Bitcoin's Academic Pedigree, *Communications of the ACM*, Vol. 60, No. 12, December 2017, pp. 36 – 45.
- [22] N. Balani, R. Hathi: *Enterprise Blockchain: A Definitive Handbook*, 1st edition, Independently published, 2017.
- [23] Z. Nehai, G. Guérard: Integration of the Blockchain in a Smart Grid Model, *Proceedings of the 14th International Conference of Young Scientists on Energy Issues – CYSENI 2017*, Kaunas, Lithuania, May 2017, pp. 127 – 134.
- [24] A. S. Musleh, G. Yao, S. M. Muyeen: Blockchain Applications in Smart Grid - Review and Frameworks, *IEEE Access*, Vol. 7, June 2019, pp. 86746 – 86757.
- [25] A Next-Generation Smart Contract and Decentralized Application Platform, Available at: <https://github.com/ethereum/wiki/wiki/White-Paper>.
- [26] M. Uday, P. D. Prasad, N. Jagadeesh Sai: Energy Trading through Blockchain, *International Journal of Scientific Research in Computer Science, Engineering and Information Technology*, Vol. 5, No. 2, March-April 2019, pp. 800 – 803.
- [27] N. Petrović: Adopting Semantic-Driven Blockchain Technology to Support Newcomers in Music Industry, *Proceedings of the 16th International Conference on Informatics and Information Technologies - CIIT 2019*, Mavrovo, North Macedonia, 2019, pp. 1 – 6.
- [28] T. R. Gruber: Toward Principles for the Design of Ontologies Used for Knowledge Sharing, *International Journal Human-Computer Studies*, Vol. 43, No. 5-6, November 1995, pp. 907 – 928.
- [29] *Semantic Web Technologies: Trends and Research in Ontology-based Systems*, Edited by J. Davies, R. Studer, P. Warren, John Wiley & Sons Ltd., Chichester, England, 2006.
- [30] Đ. Kocić, N. Petrović: Application of Android Based Devices in Analog Electric Signal Measurement, *Proceedings of the 25th Conference YulInfo 2019*, Kopaonik, Serbia, 2019, pp. 1 – 5.

- [31] I. S. Bayram, M.Z. Shakir, M. Abdallah, K. Qaraq: A Survey on Energy Trading in Smart Grid, Proceedings of the IEEE Global Conference on Signal and Information Processing – GlobalSIP, Atlanta, USA, December 2014, pp. 258 – 262.
- [32] R. Deng, Z. Yang, M.- Y. Chow, J. Chen: A Survey on Demand Response in Smart Grids: Mathematical Models and Approaches, IEEE Transactions on Industrial Informatics, Vol. 11, No. 3, June 2015, pp. 570 – 582.
- [33] Hourly Energy Consumption - Over 10 Years of Hourly Energy Consumption Data from PJM in Megawatts, Available at:
<https://www.kaggle.com/robikscube/hourly-energy-consumption>.
- [34] OpenWeather, Available at: <https://openweathermap.org/>.
- [35] B. Zhang, Z. Hao, Z. Bo: New Development in Relay Protection for Smart Grid, Protection and Control of Modern Power Systems, Vol. 1, No. 14, November 2016, pp. 1 – 7.
- [36] K. Kauhaniemi, S. Voima: Adaptive Relay Protection Concept for Smart Grids, Proceedings of the Renewable Efficient Energy II Conference, Vaasa, Finland, March 2012, pp. 1 – 10.
- [37] M. F. Kotb, M. El-Saadawi, E. H. El-Desouky: Overcurrent Protection Relay-Based Arduino Uno for FREEDM System, International Transactions on Electrical Energy Systems, Vol. 29, No. 6, June 2019, pp. 1 – 13.
- [38] M. Schvarcbacher, K. Hrabovska, B. Rossi, T. Pitner: Smart Grid Testing Management Platform (SGTMP), Applied Sciences, Vol. 8, No. 11, November 2018, pp. 1 – 23.
- [39] J. Delamare, B. Bitachon, Z. Peng, Y. Wang, B.R. Haverkort, M. R. Jongerden: Development of a Smart Grid Simulation Environment, Electronic Notes in Theoretical Computer Science, Vol. 318, No. C, November 2015, pp. 19 – 29.
- [40] N. Petrović: Approach to Dynamic Adaptivity Simulation in Fog Computing Scenarios, Proceedings of the 14th International Conference on Advanced Technologies, Systems and Services in Telecommunications – TELSIS 2019, Niš, Serbia, October 2019, pp. 58 – 61.
- [41] N. Petrović: Churn Prediction in Telco Industry Leveraging Call Center Data, Proceedings of the 6th International Conference on Electrical, Electronic and Computing Engineering - IcETRAN 2019, Silver Lake, Serbia, June 2019, pp. 845 – 850.
- [42] V. Delgado-Gomes, J. F. Martins, C. Lima, P. N. Borza: Smart Grid Security Issues, Proceedings of the 9th International Conference on Compatibility and Power Electronics – CPE, Costa da Caparica, Portugal, June 2015, pp. 534 – 538.
- [43] Z. Nehai, P.- Y. Piriou, F. Daumas: Model-Checking of Smart Contracts, Proceedings of the IEEE International Conference on Internet of Things (iThings) and IEEE Green Computing and Communications (GreenCom) and IEEE Cyber, Physical and Social Computing (CPSCom) and IEEE Smart Data (SmartData), Halifax, Canada, July 2018, pp. 980 – 987.
- [44] W. Ahrendt, R. Bubel, J. Ellul, G. Pace, R. P. Jimenez, V. Rebiscoul, G. Schneider: Verification of Smart Contract Business Logic: Exploiting a Java Source Code Verifier, Proceedings of the 8th International Conference on Fundamentals of Software Engineering – FSEN 2019, Tehran, Iran, Vol. 11761, May 2019, pp. 228 – 243.