Journal of civil engineering and transport **transEngin**

2020, Volume **2** Number **4** **ISSN 2658-1698, e-ISSN 2658-2120**

DOI: 10.24136/tren.2020.014

SIMPLIFICATION OF DEEP REINFORCEMENT LEARNING IN TRAFFIC CONTROL USING THE BONSAI PLATFORM

Michal **Skuba** [1], Aleš **Janota** [2,*]

[1] University of Žilina, Faculty of Electrical Engineering and Information Technology, Univerzitná 1, 010 26 Žilina, Slovakia, email: skuba@stud.uniza.sk, https://orcid.org/0000-0002-5453-9648

[2] University of Žilina, Faculty of Electrical Engineering and Information Technology, DCIS, Univerzitná 1, 010 26 Žilina, Slovakia, email: ales.janota@uniza.sk, https://orcid.org/0000-0003-2132-1295

* Corresponding author

Reviewed positively: 30.05.2021

Information about quoting an article:
Skuba M., Janota A. (2020). Simplification of Deep Reinforcement Learning in Traffic Control Using the Bonsai Platform. Journal of civil engineering and transport. 2(4), 191-202, ISSN 2658-1698, e-ISSN 2658-2120, DOI: 10.24136/tren.2020.014

Abstract – The paper deals with the problem of traffic light control of road intersection. The authors use a model of a real road junction created in the AnyLogic modelling tool. For two scenarios, there are three simulation experiments performed – fixed time control, fixed time control after AnyLogic-based optimizations, and dynamic control obtained through the cooperation of the AnyLogic tool and the Bonsai platform, utilizing benefits of deep reinforcement learning. At present, there are trends to simplify machine learning processes as much as possible to make them accessible to practitioners with no artificial intelligence background and without the need to become data scientists. Project Bonsai represents an easy-to-use connector, that allows to use AnyLogic models connected to the Bonsai platform - a novel approach to machine learning without the need to set any hyper-parameters. Due to unavailability of real operational data, the model uses simulation data only, with presence and movement of vehicles only (no pedestrians). The optimization problem consists in minimizing the average time that agents (vehicles) must spend in the model, passing the modelled intersection. Another observed parameter is the maximum time of individual vehicles spent in the model. The authors share their practical, mainly methodological, experiences with the simulation process and indicate economic cost needed for training as well.[1]

Key words – control, deep reinforcement learning, model, simulation, traffic

JEL Classification – L9, L91, C18, R41

INTRODUCTION

The at-grade intersections play the critical role in the urban road networks since traffic congestion permanently continues to grow. Current traffic light control systems are not ideal and often have many problems, e.g. suffer from long waiting times which causes excessive fuel consumption and the release of harmful emissions into the air. Drivers thus find themselves in stressful situations, emergency vehicles are delayed in arriving at the scene of the incident, public transport does not comply with the timetable, etc. Traffic congestion within urban road traffic networks is typically caused by the unbalanced distribution of traffic flows [1]. A lot of research has been done to improve the operational efficiency of the road junctions. The proper measures aim to optimize traffic flows and minimize the time that vehicles have to spend in the road junction area as they pass through it. Generally, there is a number of control objectives or various criteria used to optimize traffic at intersection: minimization of delays or maximization of operational efficiency, minimization of environmental impacts, maximization of safety, et al. They are solved through optimization of lane allocations, optimization of signal phasing and signal timing, application of various data models based on historical traffic flow data or traffic flow

[1] **Article was created as part of the project KEGA 008ŽU-4/2021 „Integrated Teaching for Artificial Intelligence Methods at the University of Žilina"**

predictions, their combinations, etc. [1-2]. Unfortunately, individual objectives are not coincident and may have opposite effects. In an effort to continue streamlining the whole process, new approaches are coming into play, based on the use of artificial intelligence (AI), machine learning methods and better use of increasingly available data.

The paper [3] studies traffic control method based on intelligent techniques such as agent-based approaches, fuzzy logic systems, neural network-fuzzy, multi-objective genetic algorithms, applied to intersection control. Research focused on the use of fuzzy logic in traffic control engineering has a long history, but new approaches are still emerging, often in combination with other approaches. The paper [4] improves the traffic flow on an isolated intersection with an adaptive fuzzy logic-based traffic light controller [5] also focuses on methods determining the urgency of a particular phase for adaptation of phase duration and/or phase sequencing; similarly [6] solves a multi-phase traffic light control strategy through the means of fuzzy logic. A novel approach to optimization of traffic signal timing appears in [7], discussing combination of genetic algorithms and a gradient descent like algorithm. More details about the whole family of evolutionary algorithms, including their pseudo-code forms and state-of-art applications, is available in [8]. The evolutionary programming algorithm can also help to initialize the Back Propagation neural network's parameters and then is applied to the traffic signal light control and consequent re-adjusting of control parameters [9]. Another paradigm tested in various engineering applications relates to multi-agent systems (MAS) as an organized set of agents that operate in a dynamic and shared environment. The MAS can also be used to control a set of traffic signals particularly [10] uses an adaptive fuzzy logic traffic controller to optimize control parameters of several signals at several adjacent intersections.

New data-driven approaches bring out a new research direction for all control-based systems, including those in transportation applications [11]. The survey [12] discusses the recent use of reinforcement learning (RL) techniques applied to address the problem of traffic signal control. In addition, this source also introduces the general setting of RL-based traffic signal control problem. The focus is on the so-called RL agent and the way it is defined, i.e. definitions of the reward, state, and action. The signal control problem may be either *single* (the agent controls the traffic signal under certain traffic conditions) or *multi-intersection* (with more traffic signals in the environment, controlled by one or several agents) [11-12]. Generally, the RL methods are usually classified to model-based and model-free methods [13], value-based methods and policy-based methods [14]. According to [12], currently, most RL-based methods for traffic signal control are model-free methods. Another popular method, combining the RL methods with the power of deep learning (DL) is Deep Reinforcement Learning (DRL). A complex survey of this novel and powerful AI based tool, seen in the context of the transportation research, is available in [15]. An overview of the state-of-the-art of deep learning architectures and algorithms related to network traffic control systems is provided in [16]. The paper [17] discusses a single DRL agent that uses policy gradient algorithm to manage the traffic signal of multiple intersections. A convolutional neural network solves the DRL model proposed by [18] to control the traffic light cycle. It is valuable if DRL methods can be tested on the real-world traffic data which is a case of [19]. The paper [20] both Deep Q-Learning and Policy Gradient approaches are used to optimize the traffic light timing (both phase and duration). More details related to methods of Q-learning in the given context are also available in [21], discussing optimization of the waiting time [22], examining four Q-Learning approaches with 6 different objective functions; and others. In [23] Deep Q- Learning Network is discussed with a special focus given to reward function. The paper [24] combines the self-organizing maps and the concept of RL, to adjust the traffic light period for several kinds of intersections (not all in the covered area). In [25] a system that uses object detection algorithm to sense real-time traffic scenario is introduced and the RL algorithm used to compute optimal signal timing. Another simulator introduces an autonomous traffic light control system, also based on DRL [26]. The open source Green Light District vehicle traffic simulator can serve as a testbed framework for development of a multi-agent multi-objective traffic light control system, again based on the RL approach [27]. The paper [28] proposes a novel multi-agent recurrent deep deterministic policy gradient algorithm based on deep deterministic policy gradient algorithm for traffic light control; this enhanced version considers the pedestrians as well. In [29] there is a proposal of an adaptive traffic signal controller, capable of receiving high-dimensional sensory

inputs and learning the optimal policy by directly interacting with the environment. The performance of training reinforcement learning agent may be influenced by different traffic conditions, this aspect is a scope of investigation in [30].

PROBLEM DEFINITION

The paper deals with improving road traffic conditions in one of Slovak cities (Sabinov), through AI implementation into the traffic signal control model in order to reduce an average time spent by vehicles when waiting for the green signal. We use a DRL method to solve the given optimization task. The results will be evaluated through a method of comparison of scenarios representing situation before and after DRL implementation.

1. APPLIED METHODOLOGY

The used methodology consists of the following five steps:

Step 1 – *Selection of an area of interest*: in our case we plan to improve a traffic situation at the given signal light intersection, an optimization task must be defined.

Step 2 – *Model creation*: An important aspect of the model is that it identifies itself with reality in essential matters, so that it can reflect the real states and consequences of decisions. At the same time, however, the model must not be too complex, which would result in too long DRL training and the need for high computing power. To create a model, we used the AnyLogic simulation software (www.anylogic.com).

Step 3 – *Definition of model situations - scenarios*: In this step, we create scenarios such as rush hour traffic, which results in traffic jams and long waiting times for vehicles at an intersection, i.e. scenarios for a given area, where we include problems that we want to solve or eliminate – in our case, the intention is to reduce the time spent by vehicles at the intersection.

Step 4 – *DRL training*: this step takes place after debugging the simulation and adapting the model to a given platform.

Step 5 – *DRL implementation into the model*: in this last step we need to obtain data on the performance of DRL and compare the results with other results obtained through previously implemented control algorithm(s) to see improvement.

SOFTWARE TOOLS

As mentioned above, for creation of the basic model we used the AnyLogic software. This simulation tool is able to simulate various kinds of environment from small production process, entire production lines to the entire logistics networks. It is based on the Java programming language and supports Windows, Linux and Mas OS X. Since it contains various libraries, modelling and simulation becomes quite an intuitive task. In our work, we mainly used the *Road Traffic* library. For machine learning purposes, our experiments also required to add some other libraries, particularly that needed for cooperation with the Bonsai project.

AREA OF INTEREST

Due to research objectives, where our attention focused mainly on the issue of machine learning, we have chosen a simpler road intersection as an area of our interest. Thanks to that, the demands on the complexity of the simulation and the required computing power have decreased. The top view photo of the modelled intersection (geographical coordinates: 49.101827, 21.100346) together with samples of visual model representation in AnyLogic are available in Fig. 1. In this article, the way of how the basic model was created is described very briefly. It is assumed that the reader has at least a minimum knowledge of the modelling environment. For more information, we recommend to visit the simulation tool website, where manuals, video guidelines and sample models are available. There are also many research papers discussing the topic, see e.g. [32].

MODEL DESIGN

At the very beginning, it is necessary to unify the model scales and floor plan scales for the purposes of simulation. Then we start building roads, later we will add traffic lights and add 3D models to get better visualization options. Another step consists in filling the model with agents (vehicles), and designing control logic for them. The last step covers adding controls for easy movement in simulation. The control buttons "2D", "3D" and "Štatistika" (Statistics) are used to switch between screens and facilitate orientation in simulation. Illustration of a statistic part of the model is depicted in Fig. 2. We have used the *Road Traffic* library, built-in 3D models, analytical tools and functions for model presentation. From additional libraries, we applied *Bonsai* library.

The logic of the model consists of simple blocks, which are *carSource*, *carMove* and *carDispose*. The *carSource* block serves as a source of agents (in our case vehicles). The *carMoveTo* block has the task of defining the path or as a point of reference for the

agent to move to. With the help of the *selectOutput* block, we with a certain probability divide the vehicles between the various reference points that they can reach. After reaching the reference point, all vehicles end at the finish line, in the *carDispose* block (Fig. 3 left). In an effort to add another kind of agents, pedestrians, to the model, we encountered a limit on the number of objects in the model, as we used a trial 30-day version only (obviously, the professional version has no limits in this way).

OPTIMIZATION

AnyLogic offers and makes possible different types of experiments. One of them is an optimization experiment, with which we can maximize or minimize the entered value. It uses the *OptQuest Engine*, from OptTek Systems, for optimization. A simulation engine uses various algorithms to achieve the best possible value under given simulation conditions.

In order to describe the process of adding an experiment as effectively as possible, let us show you Fig. 3. It summarizes all the necessary settings used in our experiment. In the *Objective* item, we select the *minimize* option and enter the variable we want to minimize below - in our case *root.timeInModel.mean()* which is an average time that agents (vehicles) must spend in the intersection area, or totally in the model. To access the variable, we use the path *root.variable_name*.

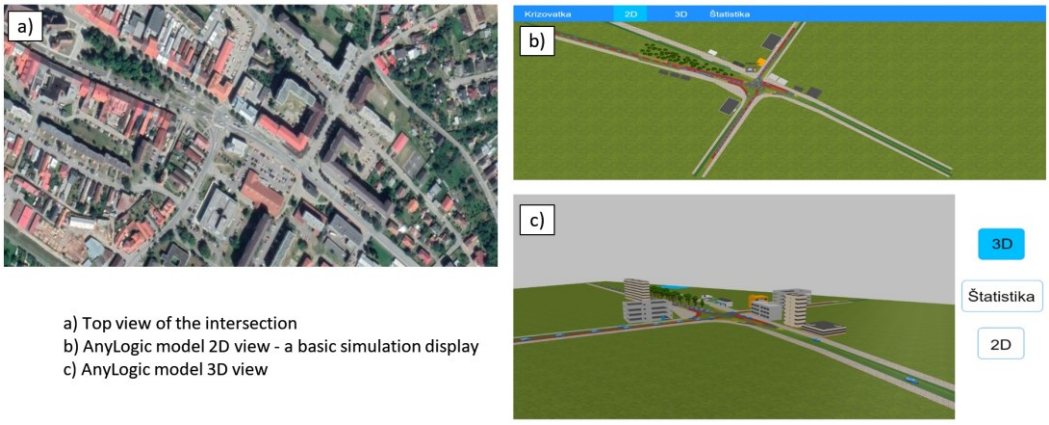

Fig. 1. Area of interest: real scenery and corresponding 2D/3D model views
(sources: a) www.google.sk/maps, b,c) own work)

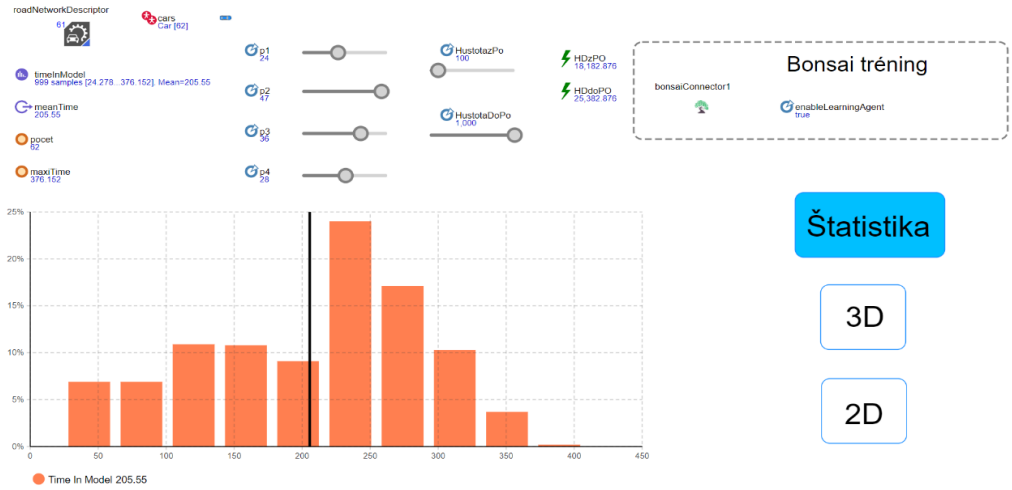

Fig. 2. Control buttons with an active statistics view – illustration (source: own work)

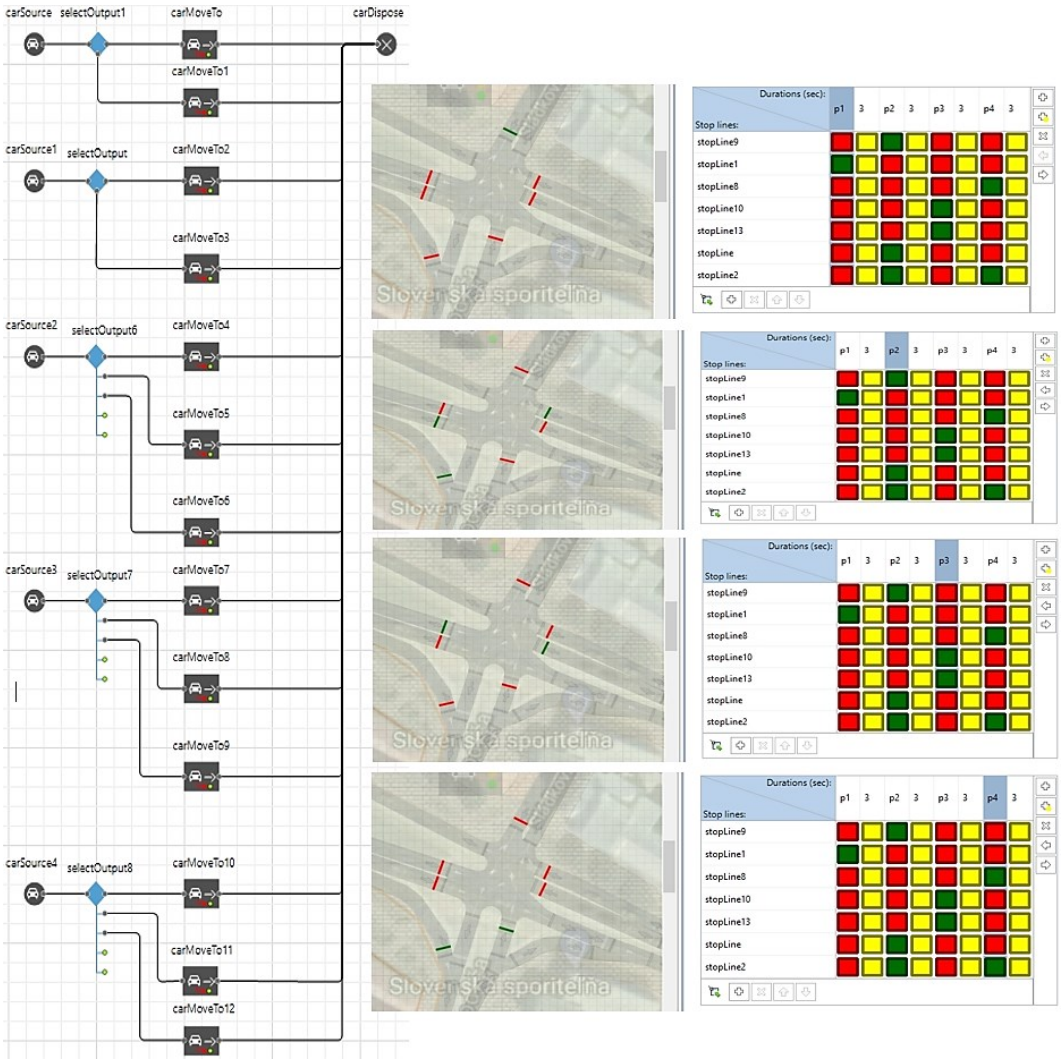

Fig. 3. Control logic for agents (left) and signal phasing (right) (source: own work)

Within the *Parameters* settings, the parameters p1 to p4 are assigned a discrete nature, the step 1 and min – max values to 5 – 50. This means that time values of the traffic lights vary from 5 seconds to 50 seconds with a step of one second. The last modification concerns the simulation time. This value was automatically copied from the simulation settings when created. It is not necessary to change it if it is satisfactory. In our case, it is set to 24 hours. We used this value with regard to the duration of the simulation, as well as the training itself. During a longer simulation, we could encounter a problem with the length of DRL training or when working with the experiments themselves. The last step (not contained in Fig. 4) is confirmation of settings; by clicking the button *Create default AI*, and running the experiment. Optimization uses processor cores for parallel simulations. The more cores, the faster the optimization. The values of the variables leading to the best results are then usable in the next simulation.

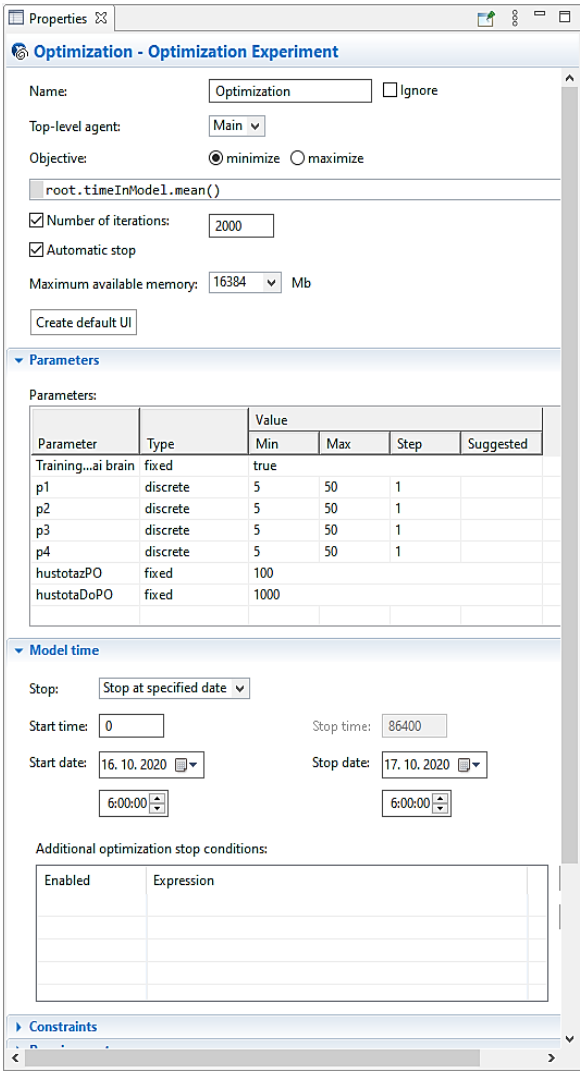

Fig. 4. Optimization settings (*source: own work*)

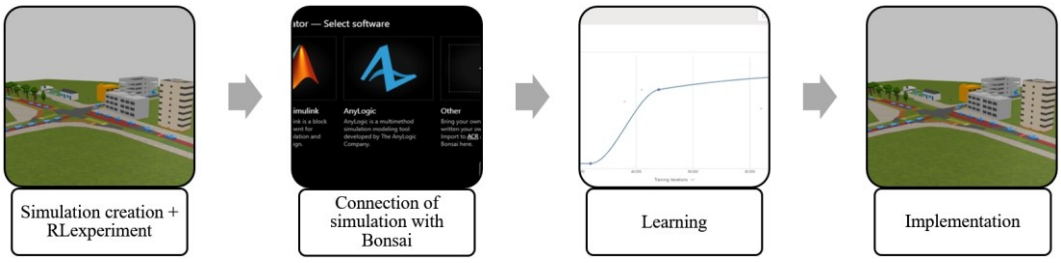

Fig. 5. The process of introducing RL for the Bonsai platform (source: own work)

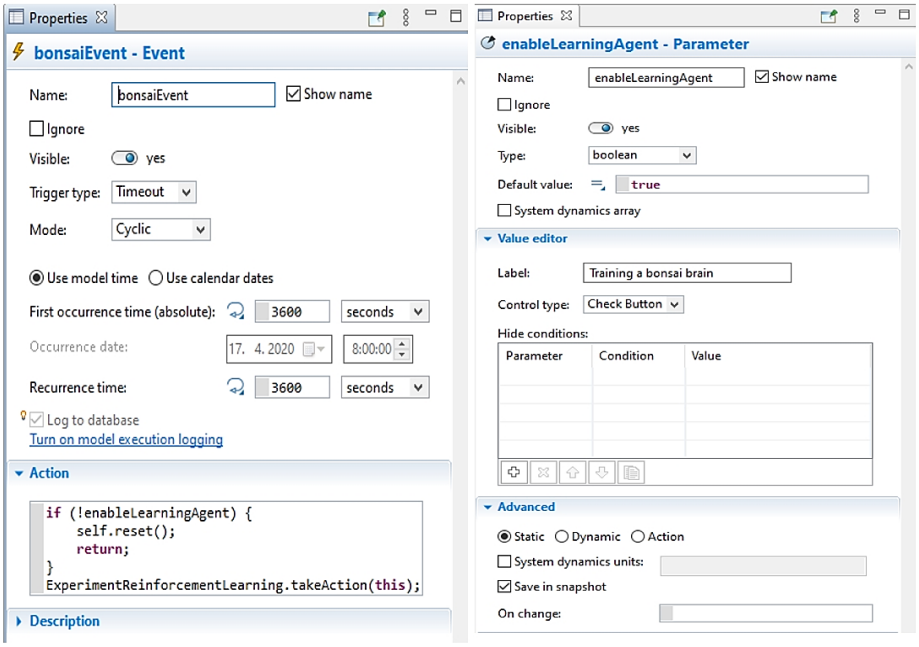

Fig. 6. Setting of *Event* and Parameters for *Bonsai*-based training (source: own work)

Another fact that we focused on during our research was to find and use a platform for easy implementation of DRL into engineering applications. Therefore, we were interested in Microsoft Bonsai (its preview version). The simplicity of implementation lies in the possibility of programming RL using *Inklink* code, developed for this platform. It uses features like *goal* or *lesson*. It means that we just need to enter the values we are interested in and what we would like to have. There are five options (minimize, maximize, avoid, drive, reach), where we determine the values we want to achieve and we can add an upper or lower limit, which the agent must not exceed. Then only the success rate of the agent is monitored in the Bonsai platform, i.e. when it reaches values in the given range. At the time of model creation, the exact of rewarding was unknown and no details were available in the documentation. Thanks to functions like *goal* or *lesson*, there is no need to invent a special reward function, which is considered one of the most difficult parts in the development of DRL. In addition, it retains the possibilities for typical DRL programming, such as the already mentioned reward function or the choice of a suitable learning algorithm. Based on Bonsai documentation from 2021, there are three options to choose from, namely APEX (DQN), PPO and SAC (the PPO algorithm was applied in our case). We did not set any hyperparameters since they are set automatically. At the time of model creation, integration with a simulation tool Simulink from MathWorks, as well as AnyLogic simulation tool, was available. It also includes the ability to create our own simulator, which we import into the ACR (Azure Container Registry) and connect to the Bonsai platform. After loading the simulator, it is possible to modify the functions by which the RL collects information about the state in the simulation, as well as the functions responsible for the individual steps through which the RL interacts with the environment in the simulation. The DRL agent only receives information about the average waiting time in the model, it receives a reward for reducing this time (reward = -timeInModel). This solution was used in the first functional prototype presented in this article, but other more appropriate rewards can be used in the future. The reward agent receives every half hour in the simulation; respectively the event after 30 minutes of simulation time stops the simulation, sends the status to the bonsai platform and receives the settings for the next half hour. Thus, changes in the simulation always happen after half an hour.

RL Implementation to the Model

First, we experimented with the *Inklink*, developed to ease programming and re-use source codes, which is a new programming language specially developed for the Bonsai platform (current version at the time of our model experiments was 2.0). The process of introducing DRL for the Bonsai platform is shown in Fig. 5. In our first experiment process, we used the version 0.2.3 later, in the second one, we tested the version 1.0.1. Adding of Bonsai library requires performing several steps; details are available at the website. The library itself is to be downloaded from the GitHub repository - https://github.com/microsoft/bonsai-anylogic/tree/master/connector. The use of 1.0.1 version significantly simplified preparation of our model for the Bonsai platform – it is available in AnyLogic 8.7 or higher. It is no longer necessary to integrate two models it is enough to generate *RLExperiment* already within our AnyLogic model. Settings of *Event* and *Parameters* for Bonsai-based training is summarized in Fig. 6.

2. Results and Discussion

The particular intersection was selected for known problems with its traffic jams during noon rush hours when a lot of people are moving from or to their offices. The main problematic directions (see Fig. 1a) are from left to right (direction to Prešov) and from right to left (direction to Lipany). Since we have had no real data , we had to rely on simulation data. Using two *events* in the model we are changing the traffic density in both principal directions from the value of 100 vehicles/hour to 1000 vehicles/hour (for rush hours from 12 p.m. to 2 p.m.) and vice versa (see Table 1). The number 1000 vehicles/hour has been reached by empirical testing of the model – by changing the traffic density in one direction from the value 5000 vehicles/hour unless the average speed becomes increasing at the end of the model route (terminated saturation). Simulation time covered the period of 24 hours.

Table 1. Traffic density values used in simulation

	Time [hours]	6 a. m.	12 p. m.	2 p. m.	12 a. m.
Traffic density [vehicles/hour]	Direction to Lipany	100	1000	1000	100
	Direction to Prešov	1000	1000	100	1000

The simulation experiment has been performed 3-times, using various time durations for light signals:

- The 1st experiment, with parameters p1 up to p4 set to a fixed value 30 s each: the average time reached at the end of experiment = 199.44 s,
- The 2nd experiment (after AnyLogic optimization): the average time of an agent (vehicle) in the model = 158.64 s,
- The 3rd experiment (with DRL implemented): the average time reached = 194.93 s.

Another parameter observed was the maximum time that an individual vehicle spent in the model. The results are summarized in Table 2.

Table 2. Comparison of various methods of traffic light signal control

Experiment	Average Time [s]	Maximum Time [s]
1: Fixed time 30 s	194.44	562.38
2: Fixed time /after optimization)	158.64	842.44
3: Dynamic time with support of DRL	194.93	522.45

The 3rd experiment represents the first version completed according the pre-set learning time values at the Bonsai platform. Instead of originally planned use of *Inklink* code option (training using the built-in function *Goal*), we had problems with, we used the *Reward* function which worked well since the very beginning. The total price for training the first version using the Bonsai platform is depicted in Fig. 7. Since it was primarily based on the student work, the cost of the simulation process was significant. It consisted of several service types - how much container instances were provided, how "rich" a container registry was, what the storage demands were, and if any log analytics was provided.

The small difference between fixed time and dynamically changing time is small due to incomplete training of AI for financial reasons. Anylogic always runs simulations in the same way unless the randomness of variables or other variables is set. So when setting the same values, we always get the same results, so we can

distinguish that it is not just about noise but about real changes using Bonsai platform control. DRL did not do as well as optimization for a similar reason - incomplete training (for financial reasons).

There was a reduction in the average time when the traffic light was controlled by the DRL in both scenarios. We've also verified that the optimization experiment in AnyLogic is only suitable for periodic and unchanged events. When changing the parameters of the model, this optimization worsened without re-examining the possibilities.

CONCLUSIONS

The article describes our first experiences with the novel approach using the Bonsai platform. presents first experiences. We created two scenarios and three different experiments for each scenario, where we tested the performance of DRL methods, as well as the performance of the optimization tool in AnyLogic. During realization of our experiments and writing of this article, the tools used and the mentioned platforms were in the process of constant improvements and modifications which sometimes complicated work with our models. In addition, we were limited in time and money. Computational power alone was not a limiting factor. The platform as well as the simulation would handle larger simulations without any significant extension of training.

To guarantee the stability and robustness of the solution, it is possible to achieve using the mentioned functions (lesson, etc.) within the Bonsai platform. Using these functions, we can change the parameters of the simulation during learning and gradually train the network to new parameters, where the result will be its general training for all the circumstances within these parameters. For future, within the Bonsai platform, we'd like to compare results obtained using the built-in *Goal* functions with our *Reward* function (not realized due to cost restrictions of our project). And similarly, we'd like to compare the results with the *Pathmind* platform whose implementation came too late for us to be able to modify our model. The novel approach based on the Bonsai platform highlights as a positive that "there is no need to invent a reward function, which is considered one of the most difficult parts in the development of artificial intelligence". Our view is that this statement is not entirely true - the remuneration function is still defined, although in some cases not directly, but through other criteria. In any case, however, the problem of designing a remuneration function is not avoided here, as would be the case e.g. when using self-supervised approaches, imitation learning or inverse learning with reward.

ABBREVIATIONS

1. **AI** – Artificial Intelligence;
2. **APEX** - A PyTorch Extension;
3. **DL** – Deep Learning;
4. **DQN** – Deep Q-Network;
5. **DRL** – Deep Reinforcement Learning;
6. **MAS** – Multi-Agent System;
7. **PPO** - Proximal Policy Optimization algorithm;
8. **RL** – Reinforcement Learning,
9. **SAC** - Soft Actor Critic algorithm.

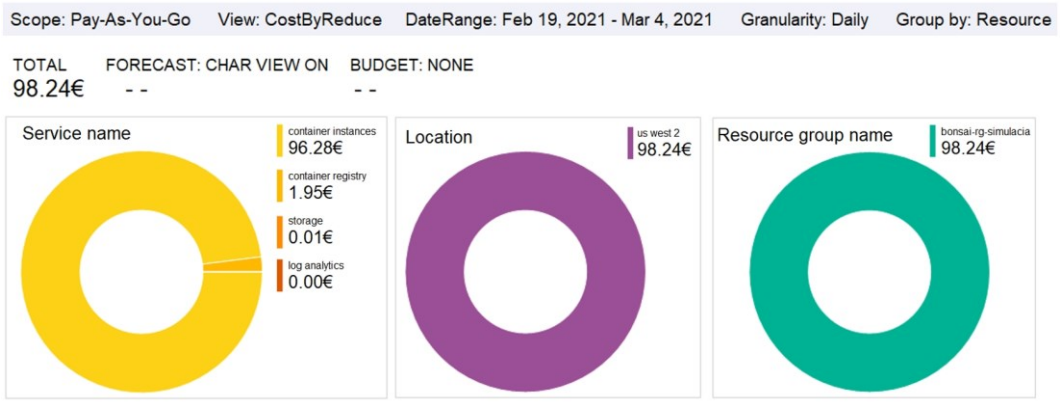

Fig. 7. The total price for DRL training in the Bonsai platform (source: own work)

UPROSZCZENIE UCZENIA SIĘ PRZEZ GŁĘBOKIE WZMOCNIENIE W ZARZĄDZANIU RUCHEM Z WYKORZYSTANIEM PLATFORMY BONSAI

Artykuł dotyczy problemu sterowania sygnalizacją świetlną na skrzyżowaniach dróg. Autorzy wykorzystują model rzeczywistego węzła drogowego utworzony w narzędziu do modelowania AnyLogic. Dla dwóch scenariuszy wykonywane są trzy eksperymenty symulacyjne - sterowanie światłami sygnalizacyjnymi o stałym czasie działania, sterowanie światłami sygnalizacyjnymi o stałym czasie działania po optymalizacji w oparciu o AnyLogic, i sterowanie dynamiczne dzięki współpracy między AnyLogic i platformą Bonsai, wykorzystując korzyści płynące z uczenia się przez głębokie wzmocnienie. Obecnie istnieją tendencje do maksymalnego upraszczania procesów uczenia maszynowego, aby były dostępne dla praktyków bez doświadczenia w zakresie sztucznej inteligencji i bez konieczności zostania naukowcami danych. Project Bonsai to łatwe w obsłudze złącze, które pozwala na korzystanie z modeli AnyLogic podłączonych do platformy Bonsai - nowatorskie podejście do uczenia maszynowego bez konieczności ustawiania hiperparametrów. Ze względu na niedostępność rzeczywistych danych eksploatacyjnych model wykorzystuje tylko dane symulacyjne, tylko z obecnością i ruchem pojazdów (bez pieszych). Problem optymalizacji polega na zminimalizowaniu średniego czasu, jaki agenci (pojazdy) muszą spędzać w modelu, mijając modelowane skrzyżowanie. Kolejnym obserwowanym parametrem jest maksymalny czas przebywania poszczególnych pojazdów w modelu. Autorzy dzielą się praktycznymi, głównie metodologicznymi, doświadczeniami związanymi z procesem symulacji oraz wskazują koszty ekonomiczne potrzebne do uczenia.

Słowa kluczowe: sterowanie, uczenie w głębokim uczeniu przez wzmacnianie, symulacja, ruch drogowy

REFERENCES

[1] Zhonghe H, Chi Z, Li W. (2015) "Consensus feedback control for urban road traffic networks", 54th Annual Conference of the Society of Instrument and Control Engineers of Japan (SICE), pp. 1413-1418. https://doi.org/10.1109/SICE.2015.7285401

[2] Zhao J, Ma W. (2021) "An alternative design for the intersections with limited traffic lanes and queuing space". IEEE Transactions on intelligent transportation systems, Vol. 22, No. 3, pp. 1473-1483. https://doi.org/10.1109/TITS.2020.2971353

[3] Wu W, Mingjun W. (2003) "Research on traffic signal control based on intelligence techniques", Proceedings of the 2003 IEEE International Conference on Intelligent Transportation Systems, vol. 1, pp. 892-896. https://doi.org/10.1109/ITSC.2003.1252078

[4] Vogel A, Oremović I, Šimić R, Ivanjko E. (2018) "Improving Traffic Light Control by Means of Fuzzy Logic", 2018 International Symposium ELMAR, pp. 51-56. https://doi.org/10.23919/ELMAR.2018.8534692

[5] Vogel A, Oremović I, Šimić R, Ivanjko E (2019) "Fuzzy Traffic Light Control Based on Phase Urgency", 2019 International Symposium ELMAR, pp. 9-14. https://doi.org/10.1109/ELMAR.2019.8918675

[6] Chai L, Shen G, Ye W. (2006) "The Traffic Flow Model for Single Intersection and its Traffic Light Intelligent Control Strategy", 2006 6th World Congress on Intelligent Control and Automation, pp. 8558-8562. https://doi.org/10.1109/WCICA.2006.1713650

[7] Yadav A, Nuthong C. (2020) "Traffic signal timings optimization based on genetic algorithm and gradient descent", 2020 5th International Conference on Computer and Communication Systems (ICCCS), pp. 670-674. https://doi.org/10.1109/ICCCS49078.2020.9118450

[8] Slowik A, Kwasnicka H. (2020) "Evolutionary algorithms and their applications to engineering problems". Neural Computing and Applications, 32, pp. 12363-12379. https://doi.org/10.1007/s00521-020-04832-8

[9] Jiang L, Li Y, Liu Y, Chen C. (2017) "Traffic signal light control model based on evolutionary programming algorithm optimization BP neural network", 2017 7th IEEE International Conference on Electronics Information and Emergency Communication (ICEIEC), pp. 564-567. https://doi.org/10.1109/ICEIEC.2017.8076629

[10] Mohammadian M. (2006) "Multi-Agents Systems for Intelligent Control of Traffic Signals", 2006 International Conference on Computational Inteligence for Modelling Control and Automation and International Conference on Intelligent Agents Web Technologies and International Commerce (CIMCA'06), pp. 270-270. https://doi.org/10.1109/CIMCA.2006.152

[11] Haydari A, Yilmaz Y. (2020) "Deep reinforcement learning for intelligent transportation systems: A survey". ArXiv, abs/2005.00935, pp. 1-22. https://arxiv.org/pdf/2005.00935.pdf

[12] Wei H, Zheng G, Gaah V, Li Z. (2021) "Recent advances in reinforcement learning for traffic signal control: A survey of models and evaluation". ACM SIGKDD Explorations Newsletter. https://doi.org/10.1145/3447556.3447565

[13] Arulkumaran K, Deisenroth M.P, Brundage M, Bharath A.A. (2017) "A brief survey of deep reinforcement learning". IEEE Signal Processing

Magazine, Special issue on deep learning for image understanding (arXiv extended version), arXiv: 1708.05866. https://doi.org/10.1109/MSP.2017.2743240

[14] Nachum O, Norouzi M, Xu K, Schuurmans D. (2017) "Bridging the gap between value and policy based reinforcement learning". 31st Conference on Neural Information Processing Systems (NIPS 2017), Long Beach, CA, USA, pp. 2772–2782. https://dl.acm.org/doi/pdf/10.5555/3294996.3295037

[15] Farazi NP, Ahamed T, Barua L, Zou B. (2020) "Deep reinforcement learning and transportation research: A comprehensive review". ArXiv abs/2020.06187, pp. 1-60. https://arxiv.org/ftp/arxiv/papers/2010/2010.06187.pdf

[16] Fadlullah ZM et al. (2017) "State-of-the-Art Deep Learning: Evolving Machine Intelligence Toward Tomorrow's Intelligent Network Traffic Control Systems", IEEE Communications Surveys & Tutorials. Vol. 19, No. 4, pp. 2432-2455, Fourthquarter 2017. https://doi.org/10.1109/COMST.2017.2707140

[17] Paul A, Mitra S. (2020) "Deep reinforcement learning based traffic signal optimization for multiple intersections in ITS", 2020 IEEE International Conference on Advanced Networks and Telecommunications Systems (ANTS), pp. 1-6. https://doi.org/10.1109/ANTS50601.2020.9342819

[18] Liang X, Du X, Wang G, Han Z. (2019) "A Deep Reinforcement Learning Network for Traffic Light Cycle Control". IEEE Transactions on Vehicular Technology. Vol. 68, No. 2, pp. 1243-1253. https://doi.org/10.1109/TVT.2018.2890726

[19] Wei H, Zheng G, Yao H, Li Z. (2018) "IntelliLight: A reinforcement learning approach for intelligent traffic light control", Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, pp. 2496-2505. https://doi.org/10.1145/3219819.3220096

[20] Coşkun M, Baggag A, Chawla S. (2018) "Deep Reinforcement Learning for Traffic Light Optimization", 2018 IEEE International Conference on Data Mining Workshops (ICDMW), pp. 564-571. https://doi.org/10.1109/ICDMW.2018.00088

[21] Rosyadi AR, Wirayuda TAB, Al-Faraby S. (2016) "Intelligent traffic light control using collaborative Q-Learning algorithms", 2016 4th International Conference on Information and Communication Technology (ICoICT), pp. 1-6. https://doi.org/10.1109/ICoICT.2016.7571925

[22] Pálos P, Huszák Á. (2020) "Comparison of Q- Learning based Traffic Light Control Methods and Objective Functions", 2020 International Conference on Software, Telecommunications and Computer Networks (SoftCOM), pp. 1-6. https://doi.org/10.23919/SoftCOM50211.2020.9238290

[23] Wu T, Kong F, Fan Z. (2019) "Road Model Design Based on Reward Function in Traffic Light Control", 2019 5th International Conference on Control, Automation and Robotics (ICCAR), pp. 407-412. https://doi.org/10.1109/ICCAR.2019.8813381

[24] Kao Y, Wu C. (2018) "A Self-Organizing Map-Based Adaptive Traffic Light Control System with Reinforcement Learning", 2018 52nd Asilomar Conference on Signals, Systems, and Computers, pp. 2060-2064. https://doi.org/10.1109/ACSSC.2018.8645125

[25] Bhave N, Dhagavkar A, Dhande K, Bana M, Joshi J. (2019) "Smart Signal – Adaptive Traffic Signal Control using Reinforcement Learning and Object Detection", 2019 Third International conference on I-SMAC (IoT in Social, Mobile, Analytics and Cloud) (I-SMAC), pp. 624-628. https://doi.org/10.1109/I-SMAC47947.2019.9032589

[26] Garg D, Chli M, Vogiatzis G. (2018) "Deep Reinforcement Learning for Autonomous Traffic Light Control", 3rd IEEE International Conference on Intelligent Transportation Engineering (ICITE), pp. 214-218. https://doi.org/10.1109/ICITE.2018.8492537

[27] Khamis M.A, Gomaa W. (2012) "Enhanced multiagent multi-objective reinforcement learning for urban traffic light control", 2012 11th International Conference on Machine Learning and Applications, pp. 586-591. https://doi.org/10.1109/ICMLA.2012.108

[28] Wu T. et al. (2020) "Multi-Agent Deep Reinforcement Learning for Urban Traffic Light Control in Vehicular Networks". IEEE Transactions on Vehicular Technology. Vol. 69, No. 8, pp. 8243-8256. https://doi.org/10.1109/TVT.2020.2997896

[29] Shabestary SMA, Abdulhai B. (2018) "Deep Learning vs. Discrete Reinforcement Learning for Adaptive Traffic Signal Control", 2018 21st International Conference on Intelligent Transportation Systems (ITSC), pp. 286-293. https://doi.org/10.1109/ITSC.2018.8569549

[30] Zeng J, Hu J, Zhang Y. (2019) "Training Reinforcement Learning Agent for Traffic Signal Control under Different Traffic Conditions", 2019 IEEE Intelligent Transportation Systems Conference (ITSC), pp. 4248-4254. https://doi.org/10.1109/ITSC.2019.8917342

[31] Skuba M. (2021) "Deep reinforcement learning use in road traffic modelling and simulation". MSc. Thesis, DCIS FEEIT University of Žilina, 52 p.

[32] Benčat G, Janota A. (2020) "Road traffic modelling based on the hybrid modelling tool AnyLogic". Journal of civil engineering and transport, Vol. 3, Issue 1, pp. 19-35. https://doi.org/10.24136/tren.2020.006