https://doi.org/10.3311/PPtr.18584

Periodica Polytechnica Transportation Engineering, 49(3), pp. 283–291, 2021

MCTS Based Approach for Solving Real-time Railway Rescheduling Problem

István Ferenc Lövétei[1*], Bálint Kővári[1], Tamás Bécsi[1]

[1] Department of Control for Transportation and Vehicle Systems, Faculty of Transportation Engineering and Vehicle Engineering, Budapest University of Technology and Economics, H-1111 Budapest, 2 Stoczek street, Hungary
[*] Corresponding author, e-mail: lovetei.istvan@kjk.bme.hu

Received: 17 May 2021, Accepted: 15 June 2021, Published online: 18 August 2021

Abstract

Solving a real-time Railway Traffic Management Problem (rtRTMP) is a challenging task for human operators. To solve the traffic situation, many factors need to be considered. Traditionally, the most critical factor is the availability of the possible routes and the relative position of the vehicles to each other. Besides, additional constraints can be found, such as the velocity, the length, and railway company regulations. The human decision-making process is essential in case of any disturbance (deviation from the pre-planned timetable). The human operator may solve this situation, but generally, the solution is not optimal. In this paper, the authors present a new method, where they consider an MCTS based algorithm to solve the traffic situation in a fast way in a given station. The performance of the algorithm is examined in two abstraction levels. The main purpose is to execute an experimental study to examine the efficiency of the MCTS based algorithms to solve railway traffic situations.

Keywords

rescheduling, rerouting, Monte Carlo Tree Search (MCTS)

1 Introduction

On the 1st of January, the program "2021 is the European Year of Rail" has started, in which the main goals are the more sustainable, smart, and safe rail transport (European Commission, 2020a).

The average punctuality of regional and local passenger services in EU27 decreased slightly from 93 % in 2015 to 90 % in 2018 (shown by Fig. 1). The average punctuality of long-distance and high-speed passenger services fell from 85 % in 2015 to 79 % in 2018. The average reliability of local and regional passenger services decreased between 2015 and 2018, with the share of cancelled services rising from 1.4 % to 1.9 %. The average reliability of long-distance and high-speed passenger services increased between 2015 and 2018, with the share of cancelled services decreasing from 1.5 % to 1.3 %. In 2018 The average punctuality of rail freight in EU27 was 60.0 % for domestic and 53.2 % for international services (shown by Fig. 2): 7.3 % of domestic and 11.0 % of international services were cancelled (European Commission, 2021a; 2021b).

The main objectives of the European "Sustainable and Smart Mobility Strategy" (European Commission, 2020a) in the rail sector are for example: greening freight transport (by doubling rail freight traffic by 2050), boosting innovation and use of data and Artificial Intelligence (AI), access to high-quality capacity maximizing the use of rail infrastructure. Rail freight needs serious boosting through increased capacity and cooperation between rail infrastructure managers, better overall management of the rail network, and the deployment of new technologies such as digital coupling and automation. New rules on rail capacity allocation are needed to execute a flexible timetable redesign and provide additional flexible train paths. The 61st objective of the strategy is to create efficient capacity allocation and traffic management rules to reduce CO_2 emissions (European Commission, 2020b).

Traffic management rules vary by country and are generally prioritized by the type of rail services (shown in Fig. 3 (European Commission, 2021b)).

Most of the trains are prioritized by Public Service Obligation (PSO). Trains with the lowest level of priority are the domestic and international freight trains. Due to the capacity problems of the infrastructure, the delays of these trains are higher than those of the passenger trains.

The priority rules vary by country since the legislation is not unique in the European Union. The new methods to

Cite this article as: Lövétei, I. F., Kővári, B., Bécsi, T. (2021) "MCTS Based Approach for Solving Real-time Railway Rescheduling Problem", Periodica Polytechnica Transportation Engineering, 49(3), pp. 283–291. https://doi.org/10.3311/PPtr.18584

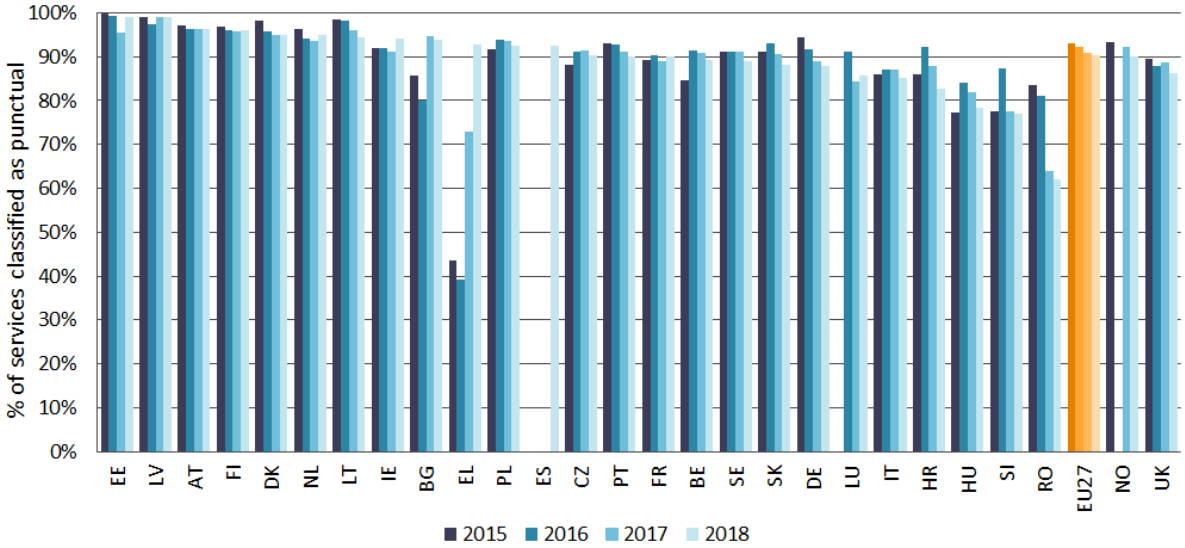

Fig. 1 Punctuality of regional and local passenger services per country in EU27, 2015–2018 (European Commission, 2021b)

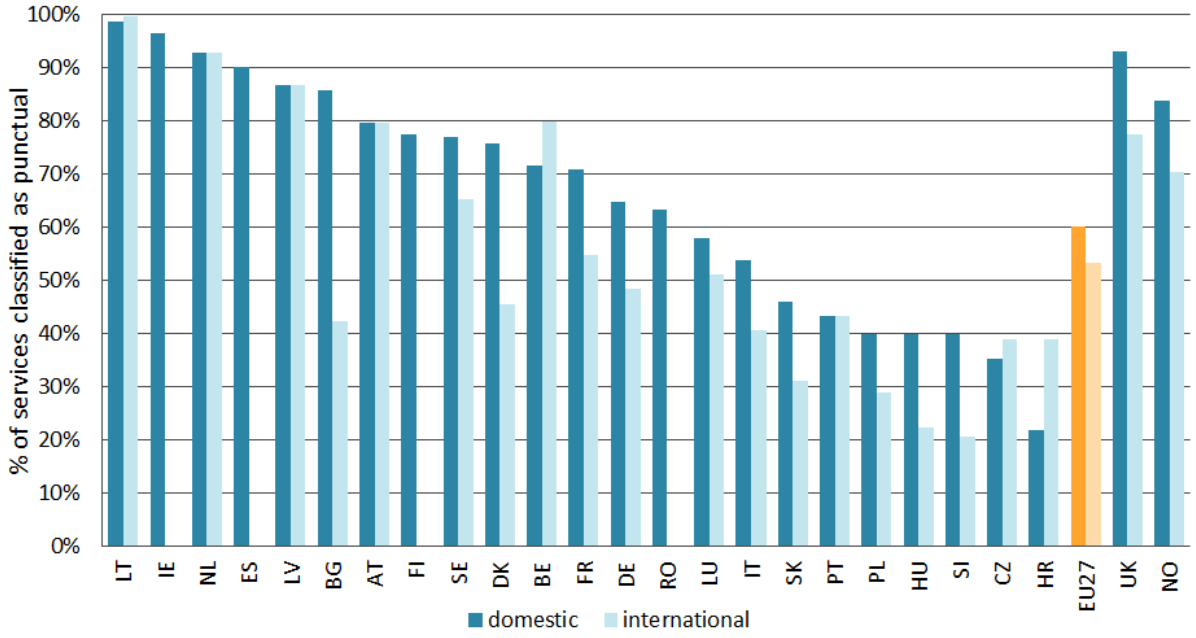

Fig. 2 Punctuality of domestic and international freight services per country in EU27, 2018 (European Commission, 2021b)

find adequate scheduling may help to make faster decisions to decrease the total travel time of freight trains. To improve the capacity and reduce the delays (together: to improve the reliability of rail transport), new operating rules are necessary. The main goal is to provide a conflict-free operation with minimized delays, where the capacity constraint of the infrastructure is a strict limitation. Nowadays, in a controlling area, human operators perform the controlling tasks with the help of some automated functions, such as the computerised route setting system. These algorithms are working correctly in a conflict-free situation. In a disturbance situation, human operators have to perform the rescheduling task.

Realization of a disturbance always means delays, which results in conflicts between trains. The traffic controlling task is still realized by train routes, and signalling aspects, with the aim of train control systems. The human operators can execute the rescheduling task, based on their experience and the regulation of the infrastructure operator. Generally, the main task is to find a rescheduled solution that means less delays and less costs. The rescheduling task also means a rerouting task, it is possible to find a new route (rerouting) for a train and also to give a new train order (rescheduling task). Other parameters may also be taken into consideration, such as the energy efficiency, but in reality, the main target is to cause less delays for the passengers. Due to

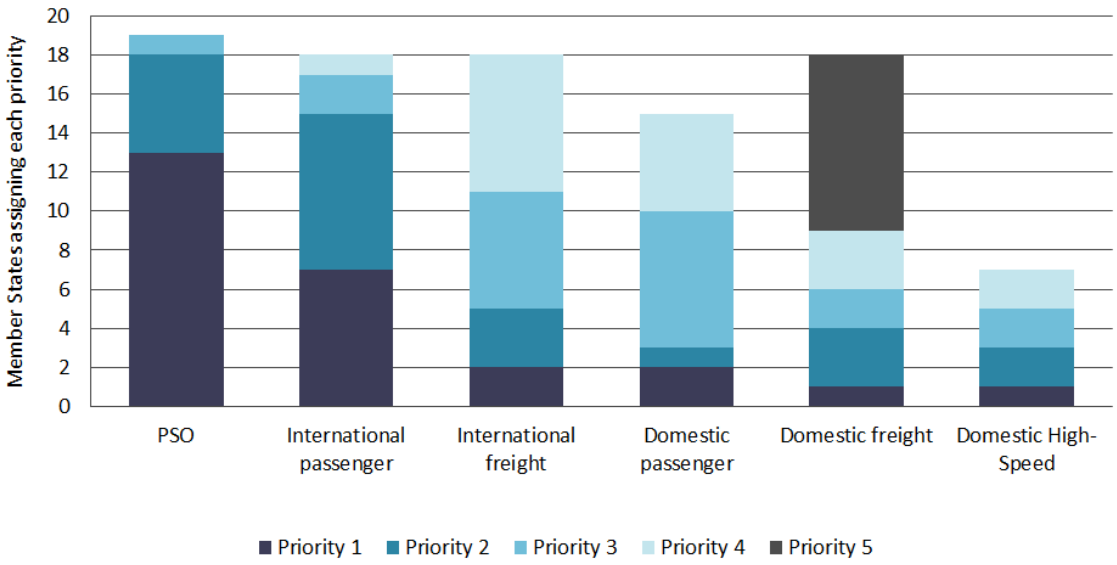

Fig. 3 Principal types of services prioritized by infrastructure managers EU27, 2018 (European Commission, 2021b)

delays, connections cannot be realized, or connections will have initial delays, that may cause other secondary delays at different locations. In an OCC (Operational Control Centre), modern controlling equipment can be in operation. Besides the safety functions - one section can be occupied just for one train - it is possible to realize other non-safety-related functions, such as timetable planning, operating the passenger information systems, etc. The central core of the system is the route setting system and to provide information through other technical systems to the train drivers (signals, train controlling systems, driver-assistant systems). Therefore, it is possible to realize other algorithms that may help the operators. The rescheduling task can be based not only on the experience of the operator but can be based on other factors, as well. One main objective is conflict detection based on the actual position and the actual speed of the vehicles (generally with the help/aim of creating? time-distance diagrams). Due to a future conflict, the rescheduling task always needs to be performed. The main principles for the rescheduling task may be the First-In, First-Out (FIFO) principle, the scheduling based on the priority (the train with the highest priority may go through first on the section, generally long-distance passenger trains) policy. If it is possible to give alternative solutions for a given traffic situation, the operator could choose between them, based on the delays, cost, and energy-efficiency. The chosen solution can be realized through the OCC. It means that the solution has to include the changed routes and signalling aspects for the trains. Also, it is possible to provide the recommended speed information to the trains through driver assistant systems.

The decision-support systems may help to solve the traffic situation in a fast and optimized way to help the railway system to be more efficient and competitive.

1.1 Problem statement

The Railway operators (dispatchers) perform the railway scheduling task many times per day, based on the predesigned daily timetable. This detailed plan contains the predefined platforms and routes. If there is no disturbance state in the controlling area, the realized platforms and train routes are the same as the predefined ones. If the capacity of the infrastructure is low, a disturbance has more severe consequences. In this case, the operators have to perform the rescheduling and rerouting task based on different rules and their experience. The new solution has to be feasible. The decision-making process has to be as short as possible. Generally, the optimization target is to minimize the total delays of passenger trains due to any disturbance. If the dispatcher likes to use other optimization targets (e.g. energy-efficiency), more complex decision-making systems will be needed. This system has to solve the rescheduling and rerouting task based on a mathematical toolbox. The search time is an essential parameter, that's why new methods and algorithms can help perform the decision-making task faster.

In our approach, the main objective is to use the MCTS algorithm to solve an rtRTMP problem. The trains have to decide at every signal, which is located at the end of each track circuit. It is a newly used method to find a feasible solution faster than other algorithms.

1.2 Related work

To solve a traffic situation, the first task is to find possible routes for each train. This problem is called a CDR (Conflict Detection and Resolution) problem. The route search algorithm has to know exactly the infrastructure. For this purpose, different algorithms use different descriptions. The rescheduling task can be performed with the aim of the "alternate-graph" method as a job-shop scheduling problem with no-wait constraints, where the operations (running through a section) are the nodes, and the arcs represent the sequence between them. Jobs means the trains, each of them consists of several operations. The goal is to define the sequence of the operations (D'Ariano et al., 2007) and (Mascis and Pacciarelli, 2002). The job-shop scheduling problem can be written and solved as a blocking parallel-machine job shop problem (Liu and Kozan, 2009). A practical application of this method is the ROMA (Railway traffic Optimization by Means of Alternative graph) real-time dispatching system (D'Ariano et al., 2008). This solution minimizes the delays.

It is possible to solve the disturbance situations with the aim of MILP (Mixed Linear Integer Programming) algorithms. The solution can be energy-optimal, and it is possible to propose new stop order for each train (Pellegrini et al., 2015). The efficiency of the MILP algorithm is verifiable by the OpenTrack simulator (Pellegrini et al., 2016). The MILP based algorithm works with the exact representation of the infrastructure and defines an objective function to minimize the total secondary delay. The MILP problem can be solved as a variable neighbourhood search metaheuristics, based on the combination of neighbourhood structures (Samà et al., 2017).

Naturally, many approaches exist for this problem. In (Farooqi et al., 2018) a cooperative solution is presented by using Nonlinear Model Predictive Control, which was further extended with Dissension based Adaptive Law in (Farooqi et al., 2019.). Answer Set Programming was used in (Abels et al., 2019), whilt in (Toletti et al., 2016), the authors propose the utilization of Resource Conflict Graphs.

A quick way to find a solution for a disturbance case is to use the greedy algorithm (Krasemann, 2010). The disadvantage of this algorithm is that it does not give the most beneficial solution.

An interesting question of the scheduling task is the size of the controlling area. Any disturbance can cause delays in a given controlling area, but the result influences the neighbouring controlling areas, as well. This problem means a multi-area railway traffic management problem (Corman et al., 2014).

It is possible to use the fuzzy-based rules to solve the railway traffic control problem and write the knowledge-based approach with Petri Nets (Fay, 2000).

1.3 Contribution

This paper proposes a search-based solution to the real-time rescheduling problem by utilizing the Monte Carlo Tree Search (MCTS) algorithm. MCTS earned tremendous success in a wide variety of control problems thanks to its versatile applicability and unique bandit-based node selection procedure that handles the exploration-exploitation trade-off. The authors also present the assessment of two different action-space abstractions that determine the branching factor of the tree and, therefore, the complexity of the optimization problem.

2 Environment

The initial problem is to solve a traffic situation without deadlock. It means that each train has to find a conflict-free path through the controlling area. Conflict-free means that the routes of the path of each train do not cross at the same time, and each train reaches its exit point successfully at the end of the simulation. Deadlock means that trains cannot move forward. In reality, the route conflicts are prevented by the help of the signalling equipment that ensures all safety functions. Each train has its own entry and exit point, and a direction. This property is essential due to the switches in the directed graph that describes the network.

In the base conception, a basic station (shown in Fig. 4) has been built. It can be foreseeable that one train cannot result in deadlock. In the case of two trains, one of them has to choose a diverted route through the diverging position of switches. In the case of the increased number of trains, the probability of a deadlock situation rises significantly. The network is described by a directed graph with the aim of a special origin-destination matrix. In this matrix, the cost (running time) of the edges and the edges between nodes can be found. The cost is directly connected to the running time, but other parameters may be taken into consideration, such as the acceleration and braking time of different types of vehicles, infrastructure parameters such as the gradient of the railway track. Generally, it means, that depending on the actual situation, the costs are not constant values, they can be computed continuously, and it is possible to update this OD matrix during the simulation if it is necessary. The running times may be positive and negative, depending on the direction. It is positive, if the direction is towards to the direction C and D, and

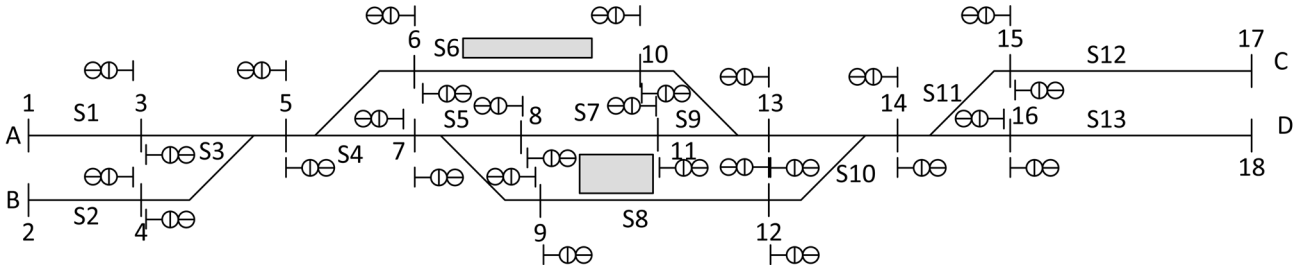

Fig. 4 Modelled network

negative otherwise. The running time has an initial value and is in contact with the possible minimum running time (computed before, with the aim of calculating? the maximum allowed speed) of a given section between given two nodes. The edges represent the track circuits, and one track circuit can be reserved just for one train at the same time. Nodes represent the decision-making points, and they can be (virtual) signals in reality. The track circuits are denoted with S. Four entry/exit points exist in the system: 1, 2, 17, and 18, the edges S1, S2, S12, and S13 are a unit. For programming purposes, this OD matrix has been split into six (three times two) arrays, that describe clearly the connections between nodes, the edges between nodes, and the running time between nodes according to the direction of traffic.

Trains may have several attributes, like entry and exit points, dwell time at the station, permitted maximal speed (that can be just equal or lower than the value associated in the OD matrix), length, etc. In the basic model, only entry and exit points are investigated. A wrong exit point means an unsuccessful simulation. In the modelled station, every simulation is run with four trains, where the initial position, the direction, and the exit point of each train are randomly generated.

To solve this randomly generated traffic situation, every train has to decide on every signal (signals are the decision-making points). The decision possibilities are generally:

- moving forward to the first direction toward the next node,
- moving forward to the second direction (just in the case where the decision-making point is directly before a junction) toward the next node,
- waiting at the signal.

During the decision-making process, it needs to be examine, that there is no other train moving toward the examined node; the given track circuit is free. Only trains directly before a signal may make a decision. The series of decisions construct a tree-based representation. The decision-making process is described in detail in Section 3.

The model and the solution were implemented in a Python environment.

3 Methodology

3.1 Monte Carlo Tree Search

Monte Carlo Tree Search (MCTS) is a best-first search algorithm that utilizes the environment's generative model to establish the tree-based representation of the control problem. MCTS's most fascinating virtues that distinguish it from other search algorithms are its value assigned to the individual nodes and how it chooses the nodes that are currently anticipated as best on a particular branch.

The value assignment is realized via random rollout. Hence the process is continued from the given state represented by the particular node according to a default policy until a terminal state is encountered. Then, the process's performance is measured through the terminal state via a chosen expression, and its value is assigned to the initial node. The Upper Confidence Bound for Trees (UCT) algorithm makes a choice between the child nodes on a particular branch. The UCT algorithm turns the exploration-exploitation dilemma into a customizable trade-off by utilizing bandit algorithms' fundamental concept (Lattimore and Szepesvári, 2020). The UCT algorithm looks as follows:

$$\overline{X_i} + 2C\sqrt{\frac{2\ln N_i}{n_i}}. \tag{1}$$

$\overline{X_i}$ represents the average value of the current node, C stands for the constant that helps control the trade-off, N_i shows how many visits the ancestor node has, and n_i indicates the number of visits of the examined node.

The introduced tree-based representation is constructed by executing the following steps of the MCTS algorithm:

- Selection: In this phase, the algorithm recursively selects the node with the highest UCT value until a leaf node is encountered.
- Expand: If the found leaf node is not a terminal state, then the algorithm populates its child nodes by utilizing the generative model of the environment and

the possible actions of the given state. If the leaf node is terminal, then the algorithm steps into the Backpropagation phase.
- Simulation: Executes a rollout from the given node, after the population of the child nodes.
- Backpropagation: This phase propagates up the leaf node's value to the root along the exact path on which the algorithm reached the leaf node.

For clarification: The MCTS algorithm is utilized as a model-based planner that builds a search tree in every time step with a limited depth where the provided iterations determine the tree's depth. After the algorithm has performed all its iterations, it recommends an action based on a policy that exploits the knowledge gathered inside the tree. In this case, the max child policy is utilized, which chooses the action that leads to the most visited child node. One of the MCTS algorithm's main benefits is that it provides a scalable solution since the number of iterations determines the horizon's size upon the actions are validated. Moreover, it is proven that the algorithm converges to the optimal solution if enough iteration is allocated (Kocsis and Szepesvári, 2006).

3.2 Cut-offs

Adding domain-specific knowledge into search trees can tremendously enhance algorithms' performance since it does not have to waste resources on branches that are already lost. MCTS has the advantage of operating with or without built-in domain-specific knowledge. In this case, MCTS is tailored to the control problem by exploiting knowledge about the trains' positions in the network. Thanks to that, our solution becomes more efficient since fewer iterations are enough to reach the same outcome.

Domain-specific knowledge is utilized in the expansion phase of the algorithm. A filter is used to ignore the population of nodes that inevitably leads to a scenario where the trains are in deadlock.

For that purpose, the network is decomposed into parts, addressed as sections in this paper. A section contains the decision-making points from junction to junction. If a section contains a decision-making point, which is a junction, then the particular section has only one element. The filter uses these sections to decide whether the current scenario is a deadlock or not.

Before populating a node, the filter goes through every train in the network and analyses each of them in the traffic scenario. The filter checks whether the analysed train has unoccupied sections branching from its current section into the train's direction. Such a section must be considered unoccupied if it does not contain any train that travels in the opposite direction as the analysed train. If there are unoccupied sections, then there is no problem, and the filter starts checking the next train until there is no unchecked train. Suppose there is no unoccupied section ahead of the analysed train. The filter has to check each train one by one that occupies the sections for the train, which is analysed in the first place. The filter's recursive call continues until it finds a train with free sections ahead, in which case the train without unoccupied sections can solve the scenario by waiting. If the trains involved in the recursion do not have any free sections ahead, then the given traffic scenario is a deadlock since, in this case, every train sharing the scenario is blocking each other's way. In other words, the filter recursively finds the trains that share the same scenario and decides whether it's a deadlock by checking whether the trains have free sections in their direction ahead.

Naturally, there are more sophisticated ways of finding deadlocks in a network. Still, this method is straightforward, easy to implement, and does not result in too much computational overhead, which is essential in search.

3.3 Abstractions of the action space

In this control problem, the algorithm's goal is to drive all the trains through the network without any conflict. At least two action-space abstractions can reach this outcome. The first is the approach where the algorithm decides the next step of all the trains in the network at once. In this approach, the tree's branching factor can change in a relatively big interval since the possible actions are all the combinations of the actions of the individual trains, which can vary based on their position. Moreover, it is hard to prioritize individual trains because only hierarchical order between the possible actions can accomplish such needs.

The second approach decides the next step for one train only at a time. Hence, it requires determining the next step for all the trains one by one to reach the same scenario as the first approach in one step. In this approach, the decision-making order can prioritize between the trains, which is efficient to realize. Unfortunately, it has some overhead in terms of computational needs compared to the first approach since much more nodes have to be populated to reach the same scenarios.

Fig. 5 displays this concern by showing both abstraction's tree for making a step for every train in a scenario where none of the trains is in a junction; hence they only

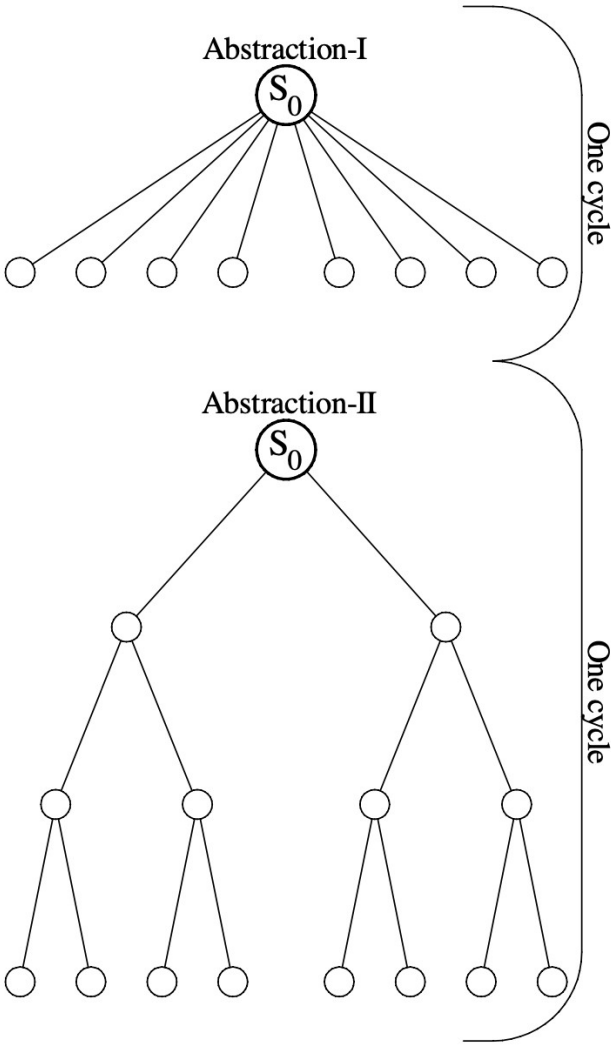

Fig. 5 The tree of both abstraction for making one step

have two actions. Abstraction-I symbolizes the first approach, while Abstraction-II illustrates the second one, and S_0 is an arbitrarily chosen initial state.

Despite all that, this approach has an important feature compared to the other. Notably, the actions have a more intuitive meaning, and the number of actions cannot be more than three in any scenario. None of them matters regarding the search. Nevertheless, our further research in this matter covers experiments with hybrid algorithms that synergize MCTS with Reinforcement Learning (RL). This action space abstraction is the one that RL can utilize efficiently. Consequently, our side goal is to assess how severely it downgrades the performance of this abstraction and investigate the possibility of such a combined method from the aspect of MCTS.

4 Results

The MCTS algorithm is implemented with both action-space abstractions. For the sake of comparability, both methods utilize the same amount of built-in domain-specific knowledge as well as the same number of iterations and exploration constant. The algorithms' performance is assessed by solving one thousand randomly generated traffic scenarios on the introduced network. In the random process that generates the traffic scenario, the trains' positions and directions are randomly chosen where deadlocks are initially excluded. For the sake of representativity, the scenarios are created with the same random seeds for both algorithms. Hence they try to solve the same scenarios. The performance is measured by the number of steps required for driving all the trains through the network on a conflict-free trajectory. This measure is calculated for the second abstraction by counting the number of cycles since a cycle is a unit where every train takes a step. Fig. 6 displays the results for the comparison.

In Fig. 6, MCTS-I strands for the MCTS with Abstraction-I and MCTS-II with Abstraction-II. The figure shows that MCTS-I has better performance as expected since it solves more episodes with fewer steps. It is essential to mention that both algorithms solved all the scenarios in a conflict-free manner.

Table 1 shows the average number of steps required for solving a traffic scenario. The difference between the approaches suggests that the deeper tree does not seriously affect the outcome. Of course, the effect on the result strongly depends on the network's complexity and the number of

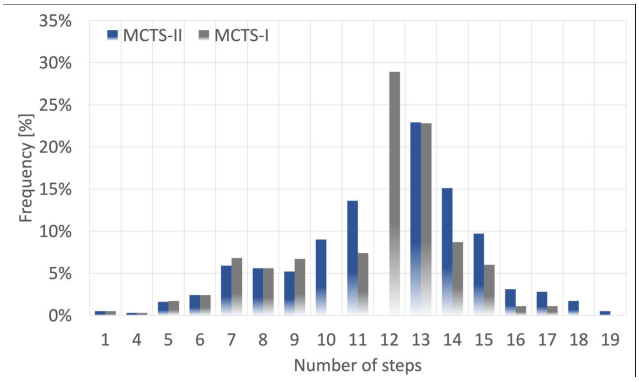

Fig. 6 The comparison of the algorithms' performances

Table 1 The comparison of the average performances

Algorithm	MCTS-I	MCTS-II
Number of steps	11.5	11.9

trains. Still, it seems feasible to take the next step to the mentioned hybrid algorithms since the results suggest that the abstraction will not degrade the performance severely.

5 Conclusion

The paper presents the search-based solution with two different abstractions to the train rescheduling problem. The results show that the MCTS algorithm can efficiently provide a scalable solution for such an application for small networks. The comparison of the abstractions points out that utilization of Abstraction-II does not result in severe deterioration of the performance. Consequently, it can be utilized in combination with Reinforcement Learning that promises the solution for much more complex networks with more trains thanks to the generalization feature of function approximators such as Neural Networks.

Acknowledgement

This research is supported by the National Research, Development and Innovation Office, under the VKE program, project 2018-1.3.1-VKE-2018-00040.

The research was supported by the Ministry of Innovation and Technology NRDI Office within the framework of the Autonomous Systems National Laboratory Program.

References

Abels, D., Jordi, J., Ostrowski, M., Schaub, T., Toletti, A., Wanko, P. (2019) "Train Scheduling with Hybrid ASP", In: 15th International Conference on Logic Programming and Nonmonotonic Reasoning, Philadelphia, PA, USA, pp. 3–17. https://doi.org/10.1007/978-3-030-20528-7_1

Corman, F., D'Ariano, A., Pacciarelli, D., Pranzo, M. (2014) "Dispatching and coordination in multi-area railway traffic management", Computers & Operations Research, 44, pp. 146–160. https://doi.org/10.1016/J.COR.2013.11.011

D'Ariano, A., Pacciarelli, D., Pranzo, M. (2007) "A branch and bound algorithm for scheduling trains in a railway network", European Journal of Operational Research, 183(2), pp. 643–657. https://doi.org/10.1016/J.EJOR.2006.10.034

D'Ariano, A., Corman, F., Pacciarelli, D., Pranzo, M. (2008) "Reordering and Local Rerouting Strategies to Manage Train Traffic in Real Time", Transportation Science, 42(4), pp. 405–419. https://doi.org/10.1287/trsc.1080.0247

European Commission (2020a) Commission Staff Working Document, "Sustainable and Smart Mobility Strategy – putting European Transport on track for the future" European Commission, Brussels, Belgium, SWD(2020) 331 final. [online] Available at: https://eur-lex.europa.eu/legal-content/EN/TXT/?uri=CELEX%3A52020SC0331&qid=1620837445352 [Accessed: 10 February 2021]

European Comission (2020b) Annex, "Sustainable and Smart Mobility Strategy – putting European Transport on track for the future", European Commission, Brussels, Belgium, COM(2020) 789 final. [online] Available at: https://eur-lex.europa.eu/resource.html?uri=cellar:5e601657-3b06-11eb-b27b-01aa75ed71a1.0001.02/DOC_2&format=PDF [Accessed: 10 February 2021]

European Commission (2021a) Report, "Seventh monitoring report on the development of the rail market under Article 15(4) of Directive 2012/34/EU of the European Parliament and of the Council", European Commission, Brussels, Belgium, COM(2021) 5 final. [online] Available at: https://ec.europa.eu/transport/sites/transport/files/com20210005-7th-rmms-report.pdf [Accessed: 10 February 2021]

European Commission (2021b) Commission Staff Working Document, "Seventh monitoring report on the development of the rail market under Article 15(4) of Directive 2012/34/EU of the European Parliament and of the Council", European Commission, Brussels, Belgium, SWD(2021) 1 final. [online] Available at: https://ec.europa.eu/transport/sites/default/files/swd20210001-7th-rmms-report.pdf [Accessed: 10 February 2021]

Farooqi, H., Incremona, G. P., Colaneri, P. (2018) "Collaborative Eco-Drive of Railway Vehicles via Switched Nonlinear Model Predictive Control", IFAC-PapersOnLine, 51(30), pp. 626–631. https://doi.org/10.1016/j.ifacol.2018.11.225

Farooqi, H., Incremona, G. P., Colaneri, P. (2019) "Railway collaborative ecodrive via dissension based switching nonlinear model predictive control", European Journal of Control, 50, pp. 153–160. https://doi.org/10.1016/j.ejcon.2019.04.005

Fay, A. (2000) "A fuzzy knowledge-based system for railway traffic control", Engineering Applications of Artificial Intelligence, 13(6), pp. 719–729. https://doi.org/10.1016/S0952-1976(00)00027-0

Krasemann, J. T. (2010) "Greedy algorithm for railway traffic re-scheduling during disturbances: a Swedish case", IET Intelligent Transport Systems, 4(4), pp. 375–386. https://doi.org/10.1049/iet-its.2009.0122

Kocsis, L., Szepesvári, Cs. (2006) "Bandit Based Monte-Carlo Planning", In: 17th European Conference on Machine Learning, Berlin, Germany, pp. 282–293. https://doi.org/10.1007/11871842_29

Lattimore, T., Szepesvári, Cs. (2020) "Bandit Algorithms", Cambridge University Press, Cambridge, UK. https://doi.org/10.1017/9781108571401

Liu, S. Q., Kozan, E. (2009) "Scheduling trains as a blocking parallel-machine job shop scheduling problem", Computers & Opeartions Research, 36(10), pp. 2840–2852. https://doi.org/10.1016/J.COR.2008.12.012

Mascis, A., Pacciarelli, D. (2002) "Job-shop scheduling with blocking and no-wait constraints", European Journal of Operational Research, 143(3), pp. 498–517. https://doi.org/10.1016/S0377-2217(01)00338-1

Pellegrini, P., Marlière, G., Pesenti, R., Rodriguez, J. (2015) "RECIFE-MILP: An Effective MILP-Based Heuristic for the Real-Time Railway Management Problem", IEEE Transactions on Intelligent Transportation Systems, 16(5), pp. 2609–2619. https://doi.org/10.1109/TITS.2015.2414294

Pellegrini, P., Marlière, R., Rodriguez, J. (2016) "A detailed analysis of the actual impact of real-time railway traffic management optimization", Journal of Rail Transport Planning & Management, 6(1), pp. 13–31. https://doi.org/10.1016/J.JRTPM.2016.01.002

Samà, M., D'Ariano, A., Corman, F., Pacciarelli, D. (2017) "A variable neighbourhood search for fast train scheduling and routing during disturbed traffic situations", Computers & Operations Research, 78, pp. 480–499. https://doi.org/10.1016/J.COR.2016.02.008

Toletti, A., De Martinis, V., Weidmann, U. (2016) "Energy savings in mixed rail traffic rescheduling: an RCG approach", In: 2016 IEEE 19th International Conference on Intelligent Transportation Systems (ITSC), Rio de Janeiro, Brazil, pp. 2430–2435. https://doi.org/10.1109/itsc.2016.7795947