Optimizing Very Large Scale ITS Applications With Fast Fitness Evaluation

Nagacharan Teja Tangirala*, Rouven Rischert*, Christoph Sommer†, and Alois Knoll*
*Chair of Robotics, Artificial Intelligence and Real-time Systems, Technical University of Munich, Munich, Germany
†TU Dresden, Faculty of Computer Science, Germany
nagacharan.tangirala@tum.de, ge47max@mytum.de, cms-labs.org/people/sommer, k@tum.de

***Abstract*—As wireless communication technology advances, the complexity of Vehicular Ad-hoc Network (VANET) simulations increases. This, however, is at odds with the need for increasingly large-scale Intelligent Transportation Systems (ITS) scenarios to satisfy the demands of increasingly Artificial Intelligence (AI) based solutions. This paper aims to demonstrate that a high-performance simplified VANET simulator can be used for fitness evaluation without loss in solution quality. As an example, we implement a Roadside Unit (RSU) deployment approach based on a genetic algorithm. *Disolv*, a simplified VANET simulator, is used as a fitness evaluation tool. To validate the solution quality, the best solutions of a few select generations are evaluated with a fully-featured ns-3 driven VANET simulation. From the comparison of fitness values, it can be observed that the values from *Disolv* allow us to predict those obtained via ns-3. Further, the execution time analysis showcases the substantial performance gains of using a more abstract VANET simulator. A 4.6-hour analysis with *Disolv* contrasts approximately 300 days with ns-3 for the given scenario and settings. Finally, the potential applications and limitations of using a simplified VANET simulator are discussed.**

***Index Terms*—Optimization, Simulation, Roadside Unit (RSU) Placement, VANET Simulation, High performance**

I. Introduction

Innovations in the smart city paradigm expand the applications of Intelligent Transportation Systems (ITS). As a result, there is a need to introduce new technologies, such as 5G and beyond, to satisfy the growing communication demands.

ITS applications are generally evaluated with the help of Vehicular Ad-hoc Network (VANET) simulators because of the expensive real-world trials. The change in paradigm and the new technologies have led to the development of several VANET simulators with even more complexity. 5GPy [1] and Simu5G [2] are some examples of recently proposed 5G system-level simulators.

Most VANET simulators, including the newly proposed 5G simulators, struggle with scalability because of the intention of high-fidelity modeling. Without the help of High-Performance Computing (HPC) resources, it is challenging to study large-scale scenarios. Researchers circumvent the issue by conducting small-scale evaluations. However, this may no longer be feasible due to the increasing involvement of Artificial Intelligence (AI), both in the protocol management as well as the ITS applications. ITS application with AI tends to require extensive data, making small-scale evaluations unreliable. Hence, large-scale VANET simulations are becoming a necessity.

In addition to scalability in terms of scenario size, some ITS applications are based on solutions requiring computationally intensive optimization approaches. A common example where optimization approaches are used is network infrastructure planning. For example, deploying Roadside Units (RSUs) or edge servers [3] over a city is a network planning study. Each possible configuration must be simulated during the parameter space exploration to determine the solution's quality. Each simulation run will take considerable time if a VANET simulator of high fidelity is used. The run-time can be extremely high at a large scale, making it difficult to traverse the parameter space and find an optimal solution.

This paper proposes an approach to solve optimization problems for very large-scale ITS applications. The high computational complexity is addressed by evaluating solution quality using a high-performance abstract VANET simulator. Initially, we quantify the performance benefits of using a high-performance VANET simulator through execution time comparisons. Also, we show a negligible deviation in the final optimal solution quality compared to a high-fidelity VANET simulator. For the sake of demonstration, we consider an RSU placement optimization study, Genetic Algorithm for Road-Side Unit Deployment (GARSUD) [4]. As a high-performance abstract VANET simulator, we select *Disolv* [5] and validate the optimal solution with ns-3.

II. Related Work

Optimization approaches are commonly employed in infrastructure planning studies such as RSU and Edge Server placement [3]. Efficient placement of RSUs is essential to support ITS applications such as content delivery [6]. Authors of [7] highlight the key role of RSUs in enabling AI applications within ITS. Literature has several applications of optimization approaches to RSU placement. Authors of [8] conducted an RSU placement study for a highway scenario to minimize the transmission delay, with the problem formulated as Integer Linear Programming (ILP). Authors of [9] carried out a similar study to determine a minimal delay deployment. Authors of [10] used ant colony optimization heuristic for RSU placement. For our analysis, we select GARSUD, a genetic algorithm based RSU placement study [4].

VANET simulators are commonly used in optimization approaches for fitness evaluation. Standalone network simulators such as ns-3 or OMNeT++ are also used to study VANET scenarios. The introduction of 5G and beyond encouraged the development of simulators dedicated to the study of 5G. Even the most recent VANET simulators, such as VSIM [11], focus on high-fidelity and protocol research. A major limitation with most VANET simulators is that they are designed for high fidelity, which results in poor scalability. As a result, the run times for a simulation run are high, making them unsuitable for fitness evaluation in optimization.

A common workaround to the scalability of VANET simulators is to reduce the scenario scale. This can be observed even in the RSU placement studies. In [4], authors employed ns-2 to simulate a small-scale scenario with a maximum of 400 vehicles. In [8], ns-2 is used for evaluation, with 50 runs per configuration. The authors dealt with scalability and the larger number of iterations by considering a small-scale scenario with a maximum of 50 vehicles. With the increasing complexity in the ITS applications, such small-scale scenarios do not represent the city-scale behavior accurately. Hence, carrying out large-scale VANET simulations is essential.

Another workaround for scalability is the custom implementation of VANET simulations. This can be observed in traffic light studies such as [12], where the authors developed a traffic density-based control system. For RSU placement, authors of [10] used a custom implementation to evaluate fitness. The custom implementation facilitated the simulation of scenarios of up to 1000 vehicles. A major issue with such custom implementations is the lack of support for reusability. Such problem-specific simulation software is often not made available to other researchers. Hence, a better approach is to employ an open-source simplified VANET simulator like *Disolv* [5], which is accessible to other researchers.

III. Workflow

A typical workflow of solving any ITS application using an optimization approach executes three phases (Initialization, Exploration, Evaluation), repeating the latter two until a satisfactory solution is reached. In more detail:

1) Initialization: An initial solution is the starting point for the optimization approach. This can either be obtained from the real world or generated randomly depending on the ITS application. Initialization must be done carefully because further exploration by the algorithm depends on the starting point. Carrying out multiple optimization runs with varying initial conditions is often recommended. Otherwise, a metaheuristic can also be used to avoid the local minima problem.

2) Evaluation: One of the primary steps in the workflow is evaluating the solution's quality. Often, an ITS application requires a complete simulation run to determine the solution quality. This is especially applicable to network planning studies at the city scale. Any infrastructure placement solution can only be evaluated by simulating at least 24 hours to capture the entire day's traffic pattern. Optimizing only for peak-hour or off-peak traffic may be insufficient. The number of possible candidate solutions to evaluate can also be significantly large depending on the optimization approach. Further, the number of agents to simulate will be considerably higher because of the city-scale scenario. As a result, the evaluation step can become a huge performance bottleneck in the context of ITS applications. Hence, making the evaluation step efficient can result in noticeable performance gains. This allows users to arrive at an optimal solution in a reasonable time.

3) Exploration: The optimization approach continues to explore possible candidates until a satisfactory solution is obtained. The exploration phase highly depends on the chosen heuristic. For example, the next candidate to explore in a genetic algorithm depends on parent selection, crossover, and mutation steps [4]. In the simulated annealing technique, the next candidate to explore is decided based on the annealing temperature scheduled by the user. Once a candidate solution is prepared, the workflow returns to the evaluation phase and the cycle continues.

In addition to the computation efforts involved in the three phases, hyperparameter tuning can be challenging. A few runs of the entire workflow must be carried out to determine appropriate hyperparameters for a given scenario, ITS application, and the selected heuristic. If the heuristic does not perform as expected, a different heuristic must be selected, and the hyperparameter tuning efforts must be repeated for the new heuristic. All of this costs significant computation efforts. Hence, optimization workflows are computationally demanding and are often carried out with the help of HPC resources.

IV. GARSUD

GARSUD follows the same optimization workflow that is described in the previous section. In this section, we describe the original implementation and the modifications.

A. GARSUD with ns-2

RSU placement involves deployment of RSUs subject to various constraints. Hence, this is a suitable problem that can be solved with the optimization workflow. Several implementations are proposed in the literature, of which we selected a genetic algorithm approach called GARSUD [4]. Genetic algorithm is a metaheuristic based on the concept of natural selection in biological evolution [13]. A population of individuals undergoes evolutionary processes, such as parent selection, mutations, recombinations, and selection. Only the individuals adapted to survive can reproduce and advance to subsequent generations. The less fit individuals automatically do not survive. A similar idea is extended to solve an optimization problem. A random solution undergoes evolution to eventually return an optimal solution.

In GARSUD, the expected optimal solution is the RSU placement. Each RSU placement is represented as an individual. In genetic algorithm terms, it is called a chromosome. The input map is divided into a grid with small cells, for example, 42x42m. Each cell is provided with a unique identifier. Given a set of RSUs to be placed, an array of their respective cell's identifiers represents the individual placement. In the evolution terminology, this acts as a chromosome, and all genetic operations are performed on it. An example of a chromosome is shown in Figure 1. For a 4-RSU scenario in a

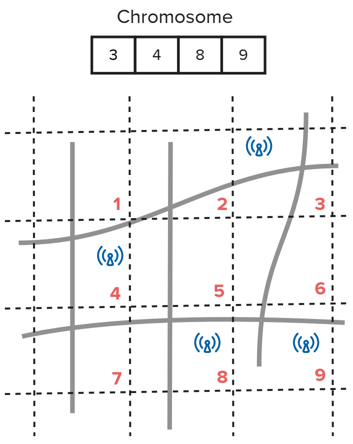

Figure 1. An example of a Chromosome for 4-RSU scenario with map divided into 9 grid cells

9-cell map, an array of 4 elements represents a chromosome. Each element in the array indicates the position of each RSU. An RSU assigned to a cell is placed randomly at one of the intersections within the cell.

The entire operation of the genetic algorithm depends on the fitness evaluation. It plays a crucial role in determining the quality of the chromosome, which in this case is the RSU placement. At the end of each generation, a predefined percentage of chromosomes with the best fitness values are selected as parents. The selected parents undergo crossover and recombination operations. A new set of individual chromosomes is generated as the next generation. Some randomization is introduced as mutation to avoid local minima issues. Finally, some portion of the parents with the best fitness values continue to survive in the coming generation. As a result, there is no chance of losing the best solution if it is found early in the evolution process.

The optimization workflow defined in Section III on the preceding page fits the overall procedure followed in the GARSUD implementation. In GARSUD, authors used ns-2 in the evaluation step of the optimization workflow. The objective focused on finding the deployment that reduces the transmission latency of emergency messages sent to the vehicles. The objective can be anything; the only criterion is that it represents good VANET conditions. Because of the performance limitations of ns-2, authors were constrained by the number of evaluations they could perform. As a result, the scenario settings are relatively simplified. The test network is 2km x 2km, and the maximum permissible transmission range is 400. The maximum density of vehicles varies to 100, 200, 300, and 400. The number of RSUs deployed in the test scenarios is set to 4 and 9. A smaller number of RSUs will reduce the possible permutations to evaluate. As a result, the number of evaluations by ns-2 is reduced, thereby arriving at a desired solution in a reasonable time. They compared the approach with other methods in literature to highlight the benefits of GARSUD. There are no other details about the execution time, and the performance impact of using ns-2 is not discussed.

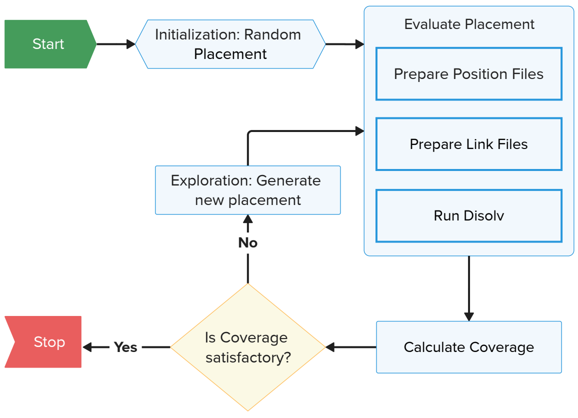

Figure 2. GARSUD Workflow for RSU Placement using *Disolv*

B. Hyperparameters

There are four main hyperparameters for genetic algorithms to control the progress of the algorithm:

- Generations: The number of iterations that the algorithm will take before termination. The higher the generations, the better the chance for the algorithm to reach an optimal solution.
- Solutions per Population: The number of placement possibilities or the offspring generated per generation. A higher number allows for a better sweep of the local neighborhood.
- Parents Mating: The number of fit candidate solutions in each generation considered for reproduction. A higher number allows for more variety in the next generation.
- Parents to Keep: The number of parent solutions with better fitness that can progress to the next generation. This prevents accidentally losing the best solution if it is found earlier in the process.

Generations and *Solutions per Population* have the highest impact on performance. Larger values increase the simulation duration and hence add to the computational load.

C. GARSUD with Disolv

In the GARSUD workflow, we replace the ns-2 with a simplified VANET simulator in the evaluation step. The goal of this paper is not to highlight the capabilities of GARSUD. Those findings can be found in the article on GARSUD [4]. Instead, the goal is to highlight that a simplified simulator enables fast fitness evaluation in optimization approaches without losing solution quality. We only use GARSUD to demonstrate our idea, and the intention is to improve the evaluation stage.

As a simplified simulator, we select *Disolv* for the reasons described in [5]. Since *Disolv* contains a simple network representation, the latency model is not as accurate as that of ns-2. Hence, the objective is modified to coverage, which can

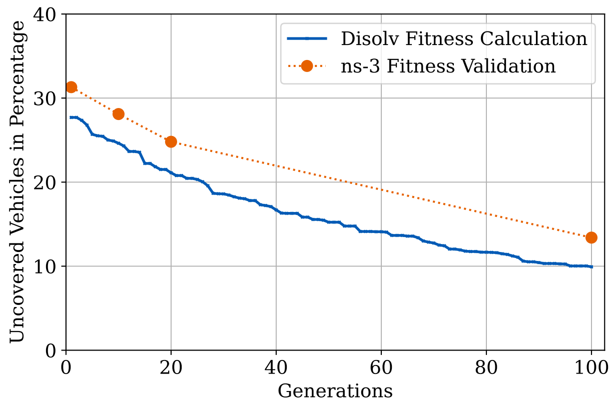

Figure 3. Fitness evaluation with *Disolv* validated by the ns-3. Fitness is evaluated with ns-3 only for generations 0, 10, 20 and 100.

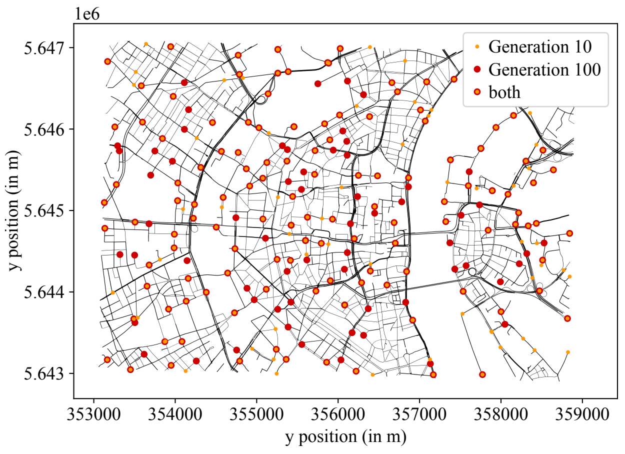

Figure 4. RSU placement solutions for generation 10 and generation 100

be obtained by running the *Disolv* simulation. Each vehicle's coverage is defined as the average percentage of the simulation duration for which it was within the transmission range of an RSU. The total coverage of the scenario is the average coverage of all the vehicles within the scenario. Due to the way coverage is defined, it can be calculated only at the end of a simulation. Hence, each possible candidate deployment is evaluated with a complete simulation run. The best solution in each generation is picked based on the coverage. The solution obtained at the last generation is expected to be optimal with the highest possible coverage. Scenario-specific tuning of hyperparameters is required to arrive at an optimal solution.

The workflow with *Disolv* contains additional steps required to make *Disolv* run. Mobility and link input files must be prepared to simulate in *Disolv* [5]. Because of the change in the positions of RSU, the preparatory steps must be repeated before each deployment evaluation. Hence, the GARSUD workflow with the introduction of *Disolv* is defined as shown in Figure 2. The preparatory steps required to run *Disolv* add additional computation load. However, we can observe in the later sections that this additional load has little impact.

V. Experiments

A sub-region of Cologne with a size of 5.5km x 4.5km is selected for the experiments. 1-hour traffic is added to the network with the help of SUMO. The simulation is run in SUMO to generate the Floating Car Data (FCD) data required for *Disolv* simulation. In [4], a transmission range of 400m is selected for the analysis. This value is relatively high for urban settings, potentially resulting in more packet failures. Hence, we reduce the transmission range to 200m. The number of RSUs to be deployed is 200.

A. Placement

Initial experiments are conducted to validate if GARSUD implementation functions as expected. The hyperparameters are selected as 100 *Generations*, 20 *Solutions per Population*, 10 *Parents Mating*, 10 *Parents to Keep*. It is important to note that scenario settings and hyperparameters are only selected for demonstration purposes. The goal is not to suggest an optimal RSU count for the scenario or the best possible combination of hyperparameters. Instead, the intention is to quantify the performance of the overall optimization workflow. Hence, the scenario and settings are arbitrarily chosen to require a noticeable computational effort.

Nevertheless, it is essential to validate the implementation. Validation can be done by observing the change in uncovered vehicles over the generations. The objective is set to calculate the coverage output for each placement, and the best value for every generation is stored. By subtracting it from 100%, we get the non-coverage, which indicates the percentage of vehicles not communicating with any RSU. This value should go down over the generations if the solution quality is improving. Indeed, the value goes down, as observed in the Figure 3. To visually validate the placement output, the road network and the positions of the RSUs are plotted in Figure 4. At generation 10, the placement solution is not ideal, and there are RSUs in smaller roads with relatively infrequent traffic. By the 100th generation, the algorithm knows to avoid roads with minimal traffic, thereby increasing the vehicles in coverage. There is potential to further optimize the placement with fewer RSUs; however, that is not our goal.

B. Validation with ns-3

One of the main goals of the experiments is to demonstrate that fast evaluation with simplified VANET simulators does not result in loss of solution quality. This can be achieved by running the entire GARSUD pipeline with a high-fidelity VANET simulator to evaluate fitness. The evolution of the fitness value with both simulators can then be compared to validate the solution quality. However, a major drawback is that the high-fidelity simulator has a high execution time. As a result, it is practically infeasible to run GARSUD with a high-fidelity simulator. Instead, we compare the best solutions of certain generations using both simulators.

For a high-fidelity simulator, we chose ns-3 along with the 5G module developed by the authors of [14]. The simulation scenario in ns-3 is designed to be similar to that of *Disolv*, and the output from the simulator is converted to a coverage value. Only the best solution of generations 1, 10, 20, and 100 are validated using ns-3. Non-coverage values from both the simulators are plotted in Figure 3. It can be observed that the fitness value progression of *Disolv* is following the same trend as that of ns-3. Due to the detailed protocol stack and the realistic models of ns-3, some packet drops were introduced in the simulation. At the time instants of these packet drops, vehicles are not considered to be in contact with the RSU, which reduces the coverage. As a result, the non-coverage value obtained from ns-3 is consistently above that obtained from *Disolv*. Hence, we can conclude that a fast VANET simulator like *Disolv* can effectively perform fast fitness evaluation without loss in solution quality.

C. Execution Time

Another goal of the paper is to quantify the performance benefits of using a simplified VANET simulator like *Disolv* for fitness evaluation. The modified workflow with *Disolv*, as shown in Figure 2, runs *Disolv* and its preparatory steps for each placement solution. Multiple placement solutions are evaluated in each generation through *Disolv* simulation. As a result, despite setting the generations as 100, the number of fitness evaluations is much higher. Further, forwarding the best parents to the next generation results in repeated solutions in each generation. This leads to redundant evaluation runs, which can be avoided to reduce computational load. To prevent such redundant evaluations, a coverage value cache is developed. The total number of times the evaluation step is called comes to 4828. With the help of the coverage cache, redundant runs are filtered out, and the remaining unique evaluations are 1566. The total time taken by the GARSUD algorithm to complete 100 generations is 15.1×10^3 seconds. From the above data, we can compute the average duration of the evaluation step to be 9.65 seconds.

The execution times for both the simulators are shown in Figure 5. The evaluation with ns-3 took 4.2 hours per candidate solution. If the GARSUD workflow was designed with ns-3, then the total execution time until the 100th generation will be 72.99×10^6 seconds, roughly 844 days. By adding a cache for repeated evaluations, the execution time can be brought down to 25.93×10^6 seconds, roughly 300 days. The selected scenario consists of 200 vehicles, and the evaluation step only simulates 1-hour duration. If the scenario is at the city scale, then the number of vehicles and RSUs are high. Further, a complete day of simulation is necessary to determine the fitness value, accounting for the traffic pattern fluctuations. This adds additional computational load to the simulator, and the execution time blows up significantly. As a result, it can be concluded that usage of ns-3 for GARSUD at a large scale is practically infeasible.

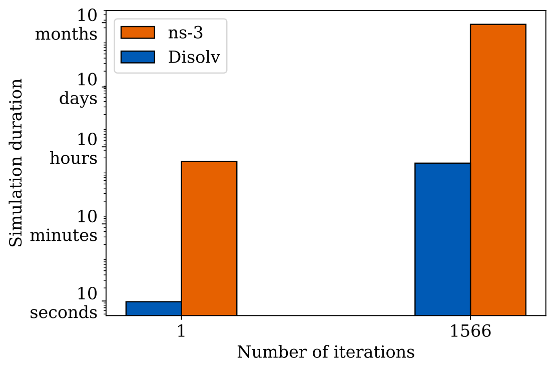

Figure 5. Execution time for Disolv and estimated time for ns-3 (note the logarithmic vertical axis)

VI. Discussion

The primary goal of this paper is to demonstrate that a simplified VANET simulator is an efficient fitness evaluation tool for optimization approaches. We selected RSU placement as an example to showcase the idea. Further, we selected GARSUD, a genetic algorithm approach, for RSU placement. Originally, the authors of GARSUD focused on reducing the transmission delay. We modified the objective to improve the coverage over the selected road network. GARSUD runs for a predefined number of generations and candidate solutions per generation. Each placement solution is evaluated by calculating the coverage based on a simulation run using a simple VANET simulator. We selected *Disolv* as a simplified VANET simulator. Finally, we validate the best solution for each generation by repeating the simulation using ns-3 with high-fidelity network models of 5G. The coverage results from ns-3 indicate that the values follow similar trends as that of *Disolv*. GARSUD algorithm took 4.6 hours to complete 100 generations. If the entire workflow of GARSUD was run with ns-3, then the runtime is extrapolated to approximately 300 days. This validates our proposal that a simplified VANET simulator is sufficient for optimization approaches in terms of solution quality while simultaneously providing fast fitness evaluation capabilities for better exploration.

A. Strengths

Using a simplified VANET simulator for fitness evaluation has multiple benefits. The obvious benefit is the high-performance capabilities compared to high-fidelity VANET simulators. With a quick turnaround, there is potential for a wider exploration of solution space. Further, hyperparameter tuning can be carried out without being affected by the computational limitations. The scale of the scenario can be expanded to city-scale, allowing for better validation of newly proposed methods.

The objective function used by the original authors of GARSUD is minimizing transmission latency. In this paper, we selected the objective of maximizing the coverage of vehicles. Any other network metrics can be used as an optimization objective. For example, the objective can be designed to reduce the average distance to the nearest RSU or server. The handover operation from one terminal (base station, RSU, server) to another can be expensive and affect service quality. The objective can be designed to reduce the number of handovers. Throughput can be another metric, and the objective can be designed to maximize the throughput. A simplified VANET simulator allows a user to carry out multiple optimization runs with different objectives. The high-performance capabilities of the simplified VANET simulator keep the runtime reasonable. Further, the best solutions can be compared to determine the suitability of an objective for a given scenario and the application.

B. Limitations

One of the limitations of using a simplified VANET simulator, besides focusing on large-scale effects (as opposed to, e.g., antenna patterns [15] or shadowing effects at intersections [16]), is the inability to detect anomalies in the VANET under overload conditions. The optimal solution in the case of GARSUD results in a placement capable of providing maximum coverage. The placement solution's performance is guaranteed to be good when the system is in a steady state. However, the characteristics of network behavior under overload, such as high latency, interference, packet failures, etc., are not detectable with simplified VANET simulators. A strategy to overcome this is to carry out additional evaluations on the final solution using a high-fidelity VANET simulator. Recently, multi-objective problems are increasingly proposed for ITS applications [17], [18]. If one of the objectives is latency, then simplified VANET simulators cannot be used for fitness evaluation.

C. Applications

In addition to the genetic algorithm approach used in this paper, several other metaheuristics are available to solve optimization problems. Simulated annealing and Tabu Search are popular meta-heuristics. They can also be analyzed to check their suitability for solving RSU placement. The RSU placement problem is only used as an example. A similar approach can be incorporated for any other ITS application that requires optimization. For example, server placement problem [3], [19] can benefit from using a simplified VANET simulator. Network slicing is another area filled with optimization approaches [20]. Any optimization approach for these ITS applications can be sped by using a simplified VANET simulator in the evaluation phase.

Deep Reinforcement learning is emerging as an alternative technique for solving optimization problems, especially the ones with multiple objectives [21]. Sometimes, a few 100s or even 1000s of episodes are required to achieve an optimal solution. If a single episode involves an entire VANET simulation run, then a simplified VANET simulator can be considered to model the environment. Similar to the optimization approaches, the final solution can be further evaluated using a high-fidelity VANET simulator.

References

[1] R. I. Tinini, M. R. P. dos Santos, G. B. Figueiredo, and D. M. Batista, "5GPy: A SimPy-based Simulator for Performance Evaluations in 5G Hybrid Cloud-Fog RAN Architectures," *Elsevier Simulation Modelling Practice and Theory*, vol. 101, May 2020.

[2] G. Nardini, D. Sabella, G. Stea, P. Thakkar, and A. Virdis, "Simu5G – An OMNeT++ Library for End-to-End Performance Evaluation of 5G Networks," *IEEE Access*, vol. 8, Jan. 2020.

[3] T. Lähderanta, T. Leppänen, L. Ruha, L. Lovén, E. Harjula, M. Ylianttila, J. Riekki, and M. J. Sillanpää, "Edge Computing Server Placement with Capacitated Location Allocation," *Elsevier Journal of Parallel and Distributed Computing*, vol. 153, Jul. 2021.

[4] M. Fogue, J. Sanguesa, F. Martinez, and J. Marquez-Barja, "Improving Roadside Unit Deployment in Vehicular Networks by Exploiting Genetic Algorithms," *MDPI Applied Sciences*, vol. 8, no. 1, Jan. 2018.

[5] N. T. Tangirala, C. Sommer, and A. Knoll, "Simulating Data Flows of Very Large Scale Intelligent Transportation Systems," in *Proceedings of the 38th ACM SIGSIM Conference on Principles of Advanced Discrete Simulation (SIGSIM-PADS 2024)*. ACM, Jun. 2024.

[6] F. Ahmed, B. Alsamani, M. Alkhathami, D. Alsadie, N. Alosaimi, B. Alenzi, and L. Nkenyereye, "Efficient content caching for 5G assisted vehicular networks," *Springer Nature Scientific Reports*, vol. 14, no. 1, Feb. 2024.

[7] D. Andreev, R. Trifonov, and M. Lazarova, "Challenges Regarding AI Integration in V2X Communication," in *2024 12th International Scientific Conference on Computer Science (COMSCI)*. IEEE, Sep. 2024.

[8] Z. Ahmed, S. Naz, and J. Ahmed, "Minimizing Transmission Delays in Vehicular Ad Hoc Networks by Optimized Placement of Road-Side Unit," *Springer Wireless Networks*, vol. 26, no. 4, Jan. 2020.

[9] H. Yu, R. Liu, Z. Li, Y. Ren, and H. Jiang, "An RSU Deployment Strategy Based on Traffic Demand in Vehicular Ad Hoc Networks (VANETs)," *IEEE Internet of Things Journal*, vol. 9, no. 9, May 2022.

[10] A. Guerna, S. Bitam, and C. T. Calafate, "AC-RDV: A Novel Ant Colony System for Roadside Units Deployment in Vehicular Ad Hoc Networks," *Springer Peer-to-Peer Networking and Applications*, vol. 14, no. 2, Oct. 2020.

[11] F. Irani, "VSIM: A New Simulation and Performance Evaluation Tool for MANET and VANET," *Springer International Journal of Information Technology*, Sep. 2024.

[12] M. Mathiane, C. Tu, P. A. Owola, and M. C. Nawej, "A SUMO Simulation Study on VANET-Based Adaptive Traffic Light Control System," in *Advances in Electrical and Computer Technologies*. Springer, 2022.

[13] C. R. Reeves and J. E. Rowe, *Genetic Algorithms—Principles and Perspectives: A Guide to GA Theory*. Springer, 2002.

[14] N. Patriciello, S. Lagen, B. Bojovic, and L. Giupponi, "An E2E simulator for 5G NR networks," *Elsevier Simulation Modelling Practice and Theory*, vol. 96, Nov. 2019.

[15] D. Eckhoff, A. Brummer, and C. Sommer, "On the Impact of Antenna Patterns on VANET Simulation," in *8th IEEE Vehicular Networking Conference (VNC 2016)*. Columbus, OH: IEEE, Dec. 2016.

[16] S. Joerer, B. Bloessl, M. Segata, C. Sommer, R. Lo Cigno, A. Jamalipour, and F. Dressler, "Enabling Situation Awareness at Intersections for IVC Congestion Control Mechanisms," *IEEE Transactions on Mobile Computing*, vol. 15, no. 7, Jul. 2016.

[17] M. Kishani, Z. Becvar, M. Nikooroo, and H. Asadi, "Joint Optimization of Communication and Storage Latencies for Vehicular Edge Computing," *IEEE Transactions on Intelligent Transportation Systems*, vol. 25, no. 6, Jun. 2024.

[18] S. Jain, V. K. Jain, and S. Mishra, "An efficient multi-objective UAV assisted RSU deployment (MOURD) scheme for VANET," *Elsevier Ad Hoc Networks*, vol. 163, Oct. 2024.

[19] A. Moubayed, A. Shami, P. Heidari, A. Larabi, and R. Brunner, "Edge-Enabled V2X Service Placement for Intelligent Transportation Systems," *IEEE Transactions on Mobile Computing*, vol. 20, no. 4, Apr. 2021.

[20] W. Attaoui, E. Sabir, H. Elbiaze, and M. Guizani, "VNF and CNF Placement in 5G: Recent Advances and Future Trends," *IEEE Transactions on Network and Service Management*, vol. 20, no. 4, Dec. 2023.

[21] J. Lu, J. Jiang, V. Balasubramanian, M. R. Khosravi, and X. Xu, "Deep Reinforcement Learning-Based Multi-Objective Edge Server Placement in Internet of Vehicles," *Elsevier Computer Communications*, vol. 187, Apr. 2022.