CoDiPy: Performance Evaluation of Vehicular Cooperative Downloading in Python

Michael Niebisch*, Daniel Pfaller†, Anatoli Djanatliev*
*Computer Networks and Communication Systems
Friedrich-Alexander-Universität Erlangen-Nürnberg, Erlangen, Germany
{michael.niebisch, anatoli.djanatliev}@fau.de
†AUDI AG, Ingolstadt, Germany
daniel.pfaller@audi.de

***Abstract*—With the increase of vehicular software, the need for efficient and cost-effective communication arises. As cellular communication alone can introduce large monetary costs, alternative communication schemes, like cooperative downloading, have been proposed. Thereby, vehicles can use direct communication technologies to collect a full set of messages. Existing network simulators can be used to evaluate the performance of such systems, however, their accurate transmission model impedes the performance. To circumvent the performance gap, we introduce Cooperative Downloading in Python (CoDiPy) to increase the simulation performance and enable large-scale and long simulation runs. We evaluate the accuracy of our results against a state-of-the-art network simulator with the same transmission parameters. CoDiPy's improves the execution time in our experiments by a factor of up to 36 when compared to a network simulator, without inducing inaccuracies.**

***Index Terms*—cooperative downloading, veins, omnet++, vehicular communication**

I. Introduction

Modern vehicles are equipped with more software-based features than ever. Besides many control units, the infotainment systems in the vehicles, as well as driving functions are implemented in software. Together with the increased reliance on digital information, the need for frequent data connections arises. As an alternative to regular visits to a workshop, the possibility of Over-the-Air (OTA) updates via cellular communication is very promising. Here, the vehicles can be used by their owners, while the car downloads the newest software or information from the servers. However, with the vast increase in software features and digital services, the required amount of data per vehicle is increasing. At the same time, the costs for data transfer are becoming an important factor for car manufacturers. The use of heterogeneous communication technologies and cooperative downloading strategies is a way, to decrease the amount of data transferred via cellular communications and is an active area of research, e.g., [1], [2]. Vehicles can exchange information between each other via direct communication technologies, and collect a full update in cooperation with other vehicles. In order to analyze the performance of such systems, simulation can be used to evaluate appropriate strategies and systems. However, traditional network simulators are ill-suited for large simulations with large simulated time spans. Their purpose is set on replicating the exact behavior of a communication technology, while a cooperative downloading system has to be evaluated in the overall performance. Their level of detail is not only unnecessary for evaluations in cooperative downloading, but also leads to excessive run-times. The goal of this work is, therefore, to evaluate the Cooperative Downloading in Python (CoDiPy) framework, used and introduced in [3], [4], for its capability of simulating Vehicle-to-Vehicle (V2V) communication realistically, while offering faster execution times. Our main contributions are:

1) Adaption of an analytical model for V2V communication.
2) Validation of achieved results of CoDiPy.
3) Reduction of execution times in comparison to a network simulation framework.

In the following Section II, we will briefly introduce the concepts of cooperative downloading, while Section III introduces the analytical model for V2V transmissions used in our simulation framework. Section V explains the basic setup of our own simulator, as well as the used simulation parameters. In Section VI, we show the results of our validation, before Section VII will conclude this work.

II. Related Work

In the context of cooperative downloading, Nandan et al. [5] introduced a gossiping mechanism, called SPAWN, for content downloading. Following them, many similar strategies and systems have been introduced, like Higuchi et al. [6] and their vehicular clouds concept. In these systems, the strategic procedure greatly influences the behavior. For example, Yao et al. [7] use linear programming for the selection of vehicles as initial starting points for the dissemination. While most authors use simulation for evaluation [6]–[9], some implement real-world testbeds like Recharte et al. [10].

978-1-6654-6975-3/22/$31.00 ©2022 IEEE

Since the applicability of existing network simulators for the evaluation of cooperative downloading schemes is limited, many works use their own frameworks for simulation [11]. The analytical modeling of real-world direct communication technologies is a constant area of research. Killat and Hartenstein [12] model the packet reception probability in a vehicular communication network, and use a Nakagami fading model, like [13]. We base our transmission calculations on these previous works and validate our modeling approach against a network simulator.

III. Analytical Model of Transmission

First, we introduce our analytical model for transmission, to establish the reception probability of a packet. To determine the path loss between a sender and receiver, we deploy the free space path loss model. It determines the receivable signal power at the receiver, dependent on the distance without interference from other signals or obstacles. The received signal power $P_{rx}(d)$ can be determined with

$$P_{rx}(d) = P_{tx} * G_{rx} G_{tx} \left(\frac{\lambda}{4\pi d}\right)^2, \tag{1}$$

where G is the gain of the antenna, λ is the wavelength, and d is the distance between sender and receiver. We can calculate the free space path loss at a reference distance of $d_0 = 1\,\mathrm{m}$ in dB with

$$\begin{aligned} L(d) = 20 * \log_{10}(f) + 20 * \log_{10}(d) \\ + 20 * \log_{10}(\frac{4\pi}{c}), \end{aligned} \tag{2}$$

where f is the center frequency of the transmission, and c is the speed of light. Further, the Log-Distance Model without shadowing can be used to describe the received signal power with

$$P_{rx}(d) = L(d_0) - 10 * \alpha * \log_{10}(\frac{d}{d_0}), \tag{3}$$

where α is the path loss exponent. If we use $\alpha = 2$, we obtain the free space path loss model from Eq. 1. The maximum achievable range for the IEEE 802.11p technology can be calculated according to [14] with

$$r_{tx} = \left[\frac{P_{tx} \cdot G_{tx} \cdot G_{rx}}{L(d_0) \cdot \gamma_{min}}\right]^{1/\alpha} \tag{4}$$

where γ_{min} denotes the minimum received signal power, depending on the chosen Modulation and Coding Scheme (MCS). We further use the Nakagami Model for a single transmitter, which can be used to determine the Packet Reception Threshold R_x. In the following, we use the analytical model described in [12], which defines the probability of receiving a message signal stronger than R_x as

$$\begin{aligned} P_R(x > R_x) &= 1 - F_d(R_x; m, \Omega) \\ &= e^{-(mR_x/\Omega)} \sum_{i=1}^{m} \frac{((m/\Omega)R_x)^{i-1}}{(i-1)!}, \end{aligned} \tag{5}$$

with $\Omega = P_{rx}$ We can use Eq. 4 in combination with Eq. 3 to determine the Packet Reception Threshold.

IV. Cooperative Downloading in Python (CoDiPy)

We have previously used and introduced our Cooperative Downloading in Python (CoDiPy) framework in [3], [4]. This python-based simulation can evaluate the impact of different data distribution strategies, as well as communication technologies, while working on a higher abstraction level than existing network simulators. For validation of our setup, we implemented the analytical models seen in Section III and introduce an additional interference parameter κ. As more communication participants lead to increased interference on the radio channel, and sometimes delayed access to the medium, we extend Eq. 3 to

$$\begin{aligned} P_{rx}(d) = L(d_0) - 10 * \alpha * \log_{10}(\frac{d}{d_0}) \\ - 10 * \kappa * \log_{10}(\frac{d}{d_0}). \end{aligned} \tag{6}$$

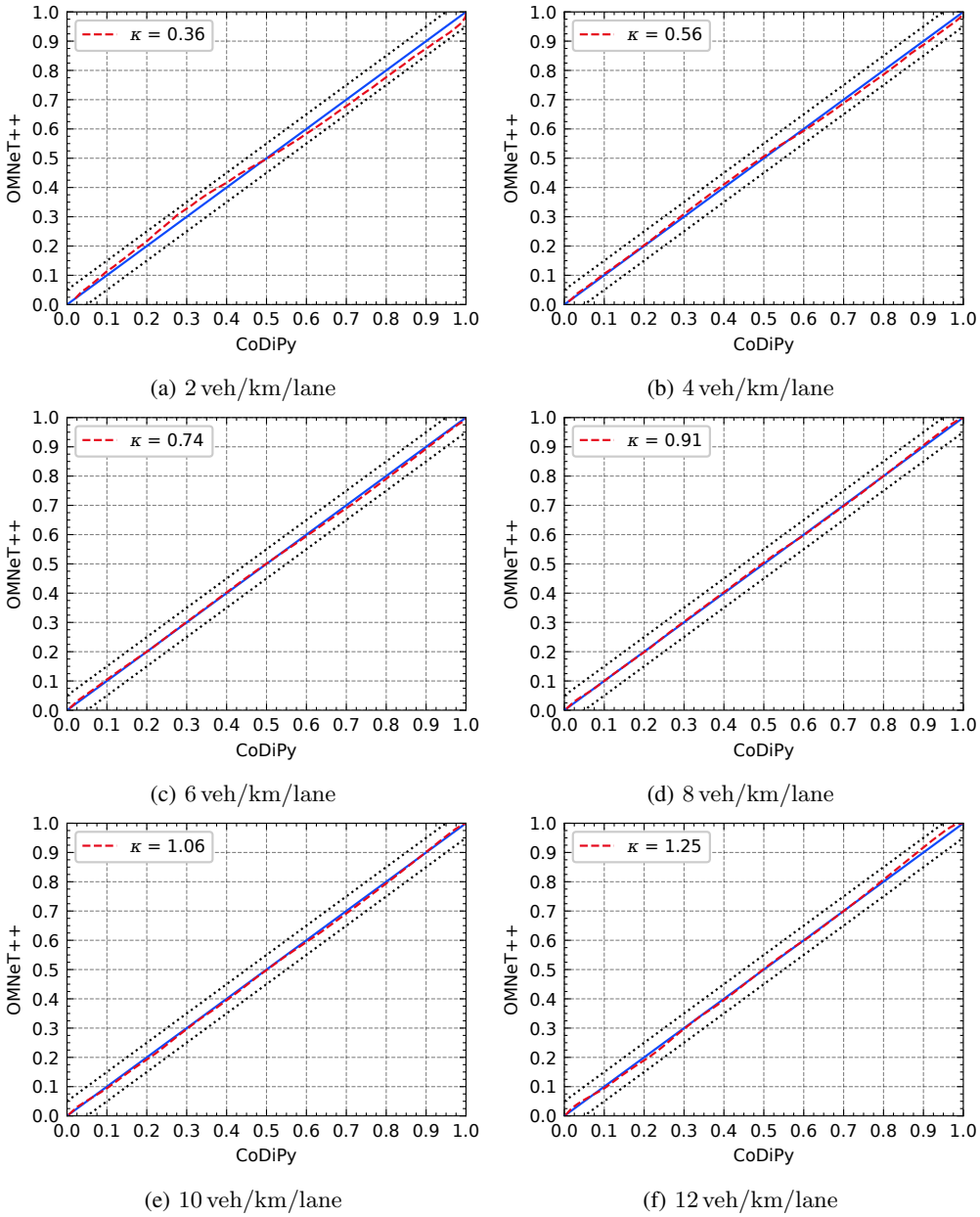
By setting κ, we can compensate for the vehicular density and transmission surroundings. This will enable us to simulate the macroscopic communication behavior, with vastly improved simulation times.

V. Simulation Setup

In order to validate the results of CoDiPy, we will use the OMNeT++ simulator [15] in combination with Veins [16] as a reference. Veins is a V2X simulation framework, that implements the IEEE 802.11p transmission standard and allows for accurate performance analysis of vehicular communication. We use the same transmission parameters (see Tab. I) like antenna gain, attenuation, and path loss for both simulators. This enables a direct comparison, while both deploy SUMO [17] for vehicular mobility and a Manhattan mobility scenario with $24\,\mathrm{km}$ of roads. We repeat all simulations ten times to achieve accurate results and use a $100\,\mathrm{MB}$ update as a medium sized file size.

VI. Simulation Results

In the following section, we compare the performance of CoDiPy and the network simulator OMNeT++. For this, we simulate the distribution of a software update using direct communication between vehicles. First, we will show the optimal choice of the interference parameter κ, depending on the vehicular density, and then indicate the achievable performance increase in execution time.



(a) 2 veh/km/lane (b) 4 veh/km/lane

(c) 6 veh/km/lane (d) 8 veh/km/lane

(e) 10 veh/km/lane (f) 12 veh/km/lane

Fig. 1: Comparison of average downloading progress of CoDiPy versus OMNeT++ for different vehicular densities

A. *Interference Compensation*

As interference influences the achievable communication performance, we account for the vehicular density by choosing a suitable interference parameter in our framework. In Fig. 1, we show the results of our comparison for different vehicular densities. The diagonal of the parity plot indicates equal progress in both simulators, averaged over 10 repetitions. The dashed lines indicate a 5 % tolerance level. We can clearly see that all simulations with CoDiPy stay inside our bounds. Fig. 2 indicates the difference between the simulators over time. The inaccuracies decrease with the increase in vehicular density, while all deviations are very low. The best fitting value of κ was chosen by minimizing the root-mean-square error (RMSE) and mean absolute percentage error (MAPE). The achieved values can be seen in Tab. II, where the RMSE is normalized between 0 and 1. For all chosen parameters, the RMSE is well below 0.1, while the MAPE is lower than 0.8 %. To indicate the dependency on the vehicular density, Fig. 3 shows a least-square fitting of κ. A linear increase can be easily seen, while the small deviations can be explained by the accuracy level of fitting κ only up to two decimal places.

TABLE I: Simulation Parameters

Parameter	Value
Simulated Area	$4\,\mathrm{km}^2$
Maximum Velocity	$50\,\mathrm{km/h}$
Communication Technology	IEEE 802.11p
MCS	2
Attenuation α	2.0
Antenna Gain G	$0\,\mathrm{dBm}$
Transmission Power P_{tx}	$20\,\mathrm{dBm}$
Noise Power	$-98\,\mathrm{dBm}$
Data Rate	$60\,\mathrm{kB/s}$
Update Size	$100\,\mathrm{MB}$
Simulation Duration	$3600\,\mathrm{s}$
Simulation Repetitions	10

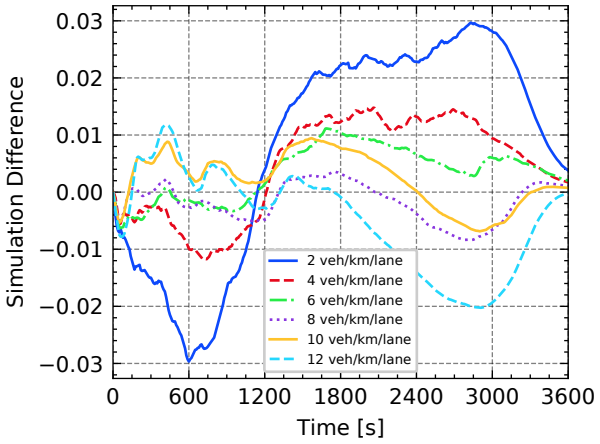


Fig. 2: Difference between CoDiPy and OMNeT++

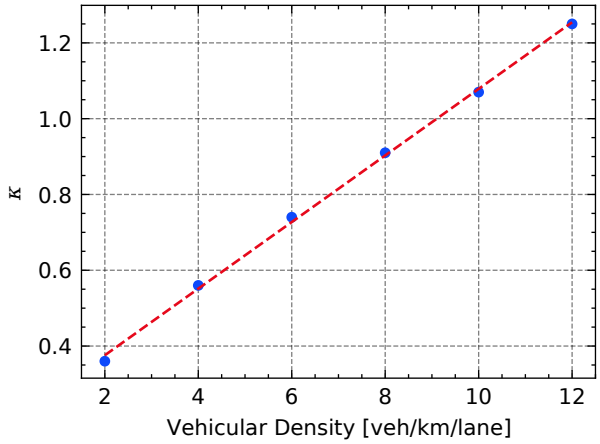


Fig. 3: Optimal selection of interference parameter κ

B. Simulation Progress

In Fig. 4, we show the exemplary progress of a CoDiPy and OMNeT++ simulation for a vehicular density of $6\,\mathrm{veh/km/lane}$, where Ψ denotes the full update. The average in both simulations progresses very similarly, while the error bars – indicating one standard deviation – shows slightly higher values for CoDiPy.

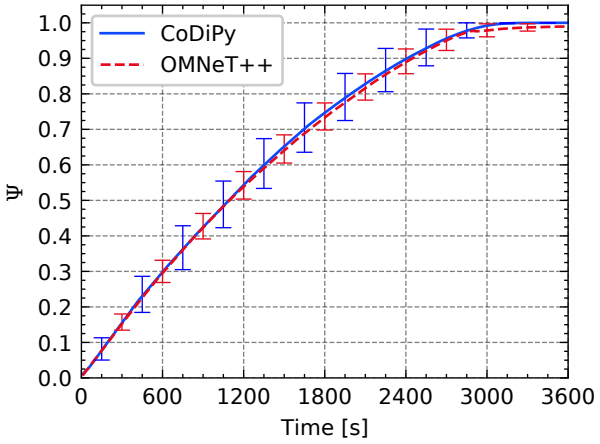


Fig. 4: Mean progress over time

C. Execution Time

The major motivation for the implementation of CoDiPy, is the reduction in execution time. In Fig. 5, we compare the average execution time for CoDiPy and OMNeT++ for simulating one hour of cooperative downloading. Factors indicate the ratio between execution time and simulated time. Even small-scaled scenarios require long execution times in OMNeT++, while a large number of vehicles decreases the usability immensely. In contrast, CoDiPy delivers small execution times even for large scenarios, while maintaining a sufficient level of accuracy. Our results show performance increases by a factor between 4 and 36, depending on the scenario. This enables fast execution and evaluation of cooperative downloading with CoDiPy.

VII. Conclusion

With the increase of vehicular software, the need for evaluating alternative communication approaches arises. We have, therefore, introduced Cooperative Downloading in Python (CoDiPy), to efficiently analyze transmission strategies and systems. We extended an analytical transmission model and validated our results with a state-of-the-art network simulator. While CoDiPy offers a macroscopic simulation of communication, the achieved results are very similar. However, our system reduces execution times from days to hours, enabling efficient evaluations. Our future work will benefit from these results, as we will use CoDiPy in combination with reinforcement learning, which requires shorter execution times than previously achievable with traditional network simulators.

Density	2	4	6	8	10	12
κ	0.36	0.56	0.74	0.91	1.06	1.25
RMSE	0.0205	0.0098	0.0057	0.0038	0.0052	0.0099
MAPE [%]	0.6468	0.1221	0.2200	0.5582	0.3772	0.7398

TABLE II: RMSE and MAPE for our optimal choice of interference κ

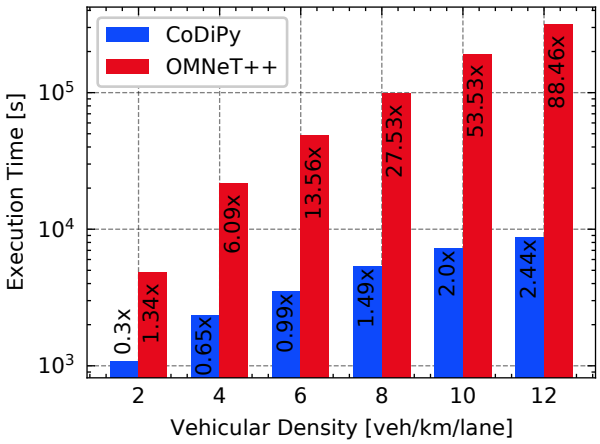


Fig. 5: Execution times for a 1 hour scenario

References

[1] L. Liu, C. Chen, T. Qiu, M. Zhang, S. Li, and B. Zhou, "A data dissemination scheme based on clustering and probabilistic broadcasting in VANETs," *Vehicular Communications*, vol. 13, pp. 78–88, 2018.

[2] F. Abbasi, M. Zarei, and A. M. Rahmani, "Fwdp: A fuzzy logic-based vehicle weighting model for data prioritization in vehicular ad hoc networks," *Vehicular Communications*, vol. 33, 2022.

[3] M. Niebisch, D. Pfaller, and A. Djanatliev, "Cooperative downloading in heterogeneous vehicular networks: A cost analysis," in *2022 International Conference on Electronics, Information, and Communication (ICEIC)*, Feb 2022, pp. 1–4.

[4] M. Niebisch, D. Pfaller, R. German, and A. Djanatliev, "Performance improvements in cooperative downloading: Encoding and strategies for heterogeneous vehicular networks," in *2022 IEEE 47th Conference on Local Computer Networks (LCN) (LCN 2022)*, Edmonton, Canada, Sep. 2022.

[5] A. Nandan, S. Das, G. Pau, M. Gerla, and M. Sanadidi, "Cooperative downloading in vehicular ad-hoc wireless networks," in *Second Annual Conference on Wireless On-demand Network Systems and Services*, 2005, pp. 32–41.

[6] T. Higuchi, R. V. Rabsatt, M. Gerla, O. Altintas, and F. Dressler, "Cooperative downloading in vehicular heterogeneous networks at the edge," in *2019 IEEE Globecom Workshops (GC Wkshps)*, 2019, pp. 1–5.

[7] H. Yao, D. Zeng, H. Huang, S. Guo, A. Barnawi, and I. Stojmenovic, "Opportunistic offloading of deadline-constrained bulk cellular traffic in vehicular dtns," *IEEE Transactions on Computers*, vol. 64, no. 12, pp. 3515–3527, Dec 2015.

[8] B. Han, P. Hui, V. A. Kumar, M. V. Marathe, G. Pei, and A. Srinivasan, "Cellular traffic offloading through opportunistic communications: A case study," in *Proceedings of the 5th ACM Workshop on Challenged Networks*, ser. CHANTS '10. New York, NY, USA: Association for Computing Machinery, 2010, p. 31–38.

[9] R. Zhang, B. Yu, and H. Krishnan, "Simulation study on collaborative content distribution in delay tolerant vehicular networks," in *2018 IEEE 88th Vehicular Technology Conference (VTC-Fall)*, 2018, pp. 1–5, ISSN: 2577-2465.

[10] D. Recharte, A. Aguiar, and H. Cabral, "Cooperative content dissemination on vehicular networks," in *2018 IEEE Vehicular Networking Conference (VNC)*, 2018, pp. 1–8, ISSN: 2157-9865.

[11] S. M. Tornell, C. T. Calafate, J.-C. Cano, and P. Manzoni, "DTN protocols for vehicular networks: An application oriented overview," *IEEE Communications Surveys Tutorials*, vol. 17, no. 2, pp. 868–887, 2015.

[12] M. Killat and H. Hartenstein, "An empirical model for probability of packet reception in vehicular ad hoc networks," *EURASIP J. Wirel. Commun. Netw.*, vol. 2009, jan 2009.

[13] X. Ma and K. S. Trivedi, "Sinr-based analysis of ieee 802.11p/bd broadcast vanets for safety services," *IEEE Transactions on Network and Service Management*, vol. 18, no. 3, pp. 2672–2686, Sep. 2021.

[14] A. Bazzi, B. M. Masini, A. Zanella, and I. Thibault, "On the performance of ieee 802.11p and lte-v2v for the cooperative awareness of connected vehicles," *IEEE Transactions on Vehicular Technology*, vol. 66, no. 11, pp. 10 419–10 432, Nov 2017.

[15] A. Varga and R. Hornig, "An Overview of the OMNeT++ Simulation Environment," in *Proceedings of the 1st International Conference on Simulation Tools and Techniques for Communications, Networks and Systems & Workshops*, ser. Simutools '08. Brussels, BEL: ICST (Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering), 2008.

[16] C. Sommer, R. German, and F. Dressler, "Bidirectionally coupled network and road traffic simulation for improved IVC analysis," *IEEE Transactions on Mobile Computing (TMC)*, vol. 10, no. 1, pp. 3–15, 2011.

[17] P. A. Lopez, M. Behrisch, L. Bieker-Walz, J. Erdmann, Y.-P. Flötteröd, R. Hilbrich, L. Lücken, J. Rummel, P. Wagner, and E. Wießner, "Microscopic traffic simulation using sumo," in *The 21st IEEE International Conference on Intelligent Transportation Systems*. IEEE, November 2018, pp. 2575–2582.