Efficient Selection of VANET Messages for Series Vehicles

Jakob Breu
Research and Development
Mercedes-Benz
Sindelfingen, Germany
jakob.breu@daimler.com

Michael Menth
Department of Computer Science
University of Tübingen
Tübingen, Germany
menth@uni-tuebingen.de

***Abstract*—In Vehicular Ad-hoc Networks (VANETs) communication among vehicles enables new advanced driver assistance systems. Cooperative Awareness Messages (CAMs) are so frequently exchanged that deployable electronic control units will not be powerful enough to process them all. Thus, most relevant CAMs need to be selected for processing.**

In this paper, we give an overview of relevance estimation function (REFs) for CAMs that assign them a relevance value based on metadata. We describe and evaluate an efficient message buffer and selection mechanism for most relevant CAMs. It efficiently decreases the relevance of buffered CAMs over time. We evaluate the results of the mechanism with regard to selection probability and message waiting time under realistic conditions. Furthermore, we study the runtime of various REFs using a prototypical implementation on a hardware evaluation platform. The results show that the proposed algorithms are feasible on close-to-production hardware as they quickly process most relevant CAMs.

I. Introduction

After decades of research, standardization, and field testing, *Vehicular Ad-Hoc Networks* (VANETs) are about to be deployed in the years ahead [1], [2]. Based on this technology, various applications with impact on driving safety, comfort, and traffic efficiency can be realized [3]. The foundation is an amendment of the 802.11 family of standards, called 802.11p [4]. Each vehicle and roadside unit that participates in a VANET broadcasts messages to its neighborhood. The Cooperative Awareness Message (CAM) is one of the standardized message formats and transmitted by vehicles up to 10 times per second [5], [6]. It contains status information about its sender and allows the receivers to create and maintain a dynamic map of their environment.

New challenges arise in the development of systems for series vehicles. In contrast to field test and prototype vehicles, the hardware has to meet strict requirements regarding cost, robustness and size. However, we showed in a study that depending on the road topology, vehicle density, movement patterns and technology penetration rate, high rates of CAMs may be received [7], [8]. This should be taken into account for the design of the device processing the CAMs. We found that one architectural key element for that device is the filtering of relevant messages, which requires an efficient function to estimate the relevance of all incoming messages. We proposed such functions in [9], [10]. A comparison of our and other relevance estimation functions (REFs) showed that the quality of their results depends on the chosen scenarios and applications [11].

In this paper, we propose an efficient message buffering and selection system. Messages are buffered and the most relevant message according to a given REF is chosen to be processed next. If a new message arrives in the presence of a full buffer, the message is dropped if it is less relevant than any other message in the buffer; otherwise, it is accommodated and a less relevant message is dropped from the buffer. Thereby, only most relevant messages are processed. This simple approach is complicated by the fact that message relevance changes over time and messages may become obsolete through consecutive messages from the same sender. We implemented this message selection and buffering concept and the REFs on realistic hardware. We measured the runtime performance of various implemented REFs and analyzed relevance distributions and waiting times of queued messages. To that end, we stressed the hardware with realistic overload scenarios generated by our simulation tool chain.

The remainder of this paper is organized as follows. Section II gives an overview of proposed REFs. Section III describes the message buffering and selection system. In Section IV the evaluation environment is introduced. In Section V, the REFs' computation times are presented and Section VI discusses performance results of the selection mechanism. Section VII concludes this work.

II. Relevance Estimation Functions

The computation of a CAM's relevance on an electronic control unit needs to satisfy multiple constraints: feasibility with available hardware resources, consideration of available information, and the interests of applications.

In the following, we present several REFs that are tailored for safety applications where the distance between vehicles plays an important role. Parameters recommended for these REFs are compiled in Table I. Some of these REFs were originally intended for message routing and forwarding algorithms [12], [13], [14].

Parameter	Value	Parameter	Value
$d_{\min}$	10 m	α	0.015
γ	0.3821	$\Delta d_{\max}$	1000 m
$t_{\max}$	10 s	β	0.15
		$\Delta t_{\max}$	10 s

TABLE I
RECOMMENDED PARAMETERS FOR VARIOUS REFS.

A. Distance-Based Relevance Estimation

In the literature a message's relevance is often solely based on the Euclidean distance $d(t_{\text{rcv}})$ between the sender and receiver at the time of the message reception t_{rcv} [15]. We call this approach *distanceRE* and compute the relevance by

$$R_{\text{distance}} = \frac{1}{\max(d_{\min}, d(t_{\text{rcv}}))}, \qquad (1)$$

where the distance $d_{\min}$ defines the radius of the direct vicinity of the receiving vehicle within which all vehicles have maximum relevance.

B. Movement-Based Relevance Estimation

Due to the dynamic nature of road traffic, it makes sense to respect the movement of vehicles for relevance estimation of CAMs. Thus, speed and heading of the sender are included, which are available in any CAM. In [9] we proposed to calculate the relevance by

$$R_{\text{general}} = \max_{t_{\text{rcv}} \le t \le t_{\text{rcv}} + t_{\max}} \left(\frac{1}{\max(d_{\min}, d(t))} \cdot \frac{1}{(\frac{t}{\text{s}} + 1)^{\gamma}} \right), \qquad (2)$$

where $d(t)$ is the distance between sending and receiving vehicle at time t given in seconds. The function maximizes the relevance while predicting the distance $d(t)$ over a limited time interval, i.e., $t_{\text{rcv}} \le t \le t_{\text{rcv}} + t_{\max}$. A penalty term reduces the relevance over time, parameter γ is used for that purpose. For the distance prediction, we proposed two different extrapolation functions for the movement of the vehicles which have different complexity.

1) Static Movement Extrapolation: We assume that both sending and receiving vehicles move with constant speed and heading given at t_{rcv} and call this approach *staticRE*. Its formula is given in [9]. It disregards the fact that vehicles change their heading and speed. As a result, the relevance values based on static movement extrapolation may significantly underestimate relevances if the sender approaches the receiver faster than extrapolated.

2) Dynamic Movement Extrapolation: The dynamic movement extrapolation (*dynamicRE*) considers changes of vehicle movement in both speed and heading [10]. As we do not know how these changes will take place, we assume a movement of the sender towards the receiver such that the relevance is maximized.

Unlike the other presented REFs, the dynamicRE requires an iterative algorithm for relevance computation [10]. It iteratively changes the speed and heading of the sending vehicle such that it moves towards the receiving vehicle. Therefore, its computation takes significantly longer to execute than the one of other approaches. To enable fast execution, we implemented this algorithm as a characteristic map. We calculated the relevances for variations of relative position (Δx, Δy), relative speed (Δv) and absolute speed of the sending vehicle (v_{S}), and stored them in a four-dimensional matrix. An approximation of the relevance value can be determined by interpolation of the values from this matrix. We use linear interpolation for efficiency reasons [16]. This specific implementation is called *mappedRE*.

C. Encounter Probability

The encounter probability approach was proposed for message forwarding (*encounterRE*) in [14]. However, it may also be used for message selection purposes. It assumes static vehicle movement and calculates the shortest distance Δd between sending and receiving vehicles in the near future by

$$R_{\text{EP}} = \frac{1}{\alpha \cdot \min(\Delta d, \Delta d_{\max}) + \beta \cdot \min(\Delta t, \Delta t_{\max}) + 1}. \qquad (3)$$

The parameter Δt is the time when Δd is reached, α and β are weights, and thresholds $\Delta d_{\max}$ and $\Delta t_{\max}$ limit Δd and Δt.

III. SELECTION MECHANISM

In this section we develop a mechanism that buffers messages and periodically selects the most relevant one according to a given REF for processing. We propose to use a reasonably sized priority queue to store CAMs in the order of their relevance and to dequeue most relevant messages for processing. Then we improve this simple idea by two features. First, we provide a data structure to efficiently avoid that multiple messages from the same sender are stored in the queue. Second, we add a scalable mechanism for message aging, so that younger messages may be preferred to older messages even though they are less relevant at their arrival.

The overall mechanism presented in this section combines well-known concepts like priority queues and hash tables with an aging mechanism to meet the specific requirements of an automotive environment and its state-of-the-art driver assistance systems.

A. A Priority Queue

We assume that when messages are received, their relevance is computed with a REF and tagged to them. We organize a message buffer as priority queue whose items are sorted according to the tagged relevance. We further assume that messages are processed at rate r_{process} and implement a thread that periodically polls the queue with a period of $\frac{1}{r_{\text{process}}}$. If a message waits longer than 1 s, it can be considered outdated because CAMs are generated at least once per second [5]. Therefore, the queue size $Q_{\max}$ can be limited such that the least relevant message in the queue does not wait longer than 1 s for processing if no more relevant messages arrive until then. Therefore, $Q_{\max} = r_{\text{process}} \cdot 1$ s is sufficient. Upon arrival of a message, the message is inserted into the sorted queue according to the tagged relevance if the queue is not full. If the queue is full and the tagged relevance of the least relevant item in the queue is less relevant than the new message, that least relevant message is deleted from the queue and the new message is inserted. Otherwise, the new message is discarded.

The data structure for the priority queue has to provide efficient methods for insertion of new messages, retrieval and deletion of the least and most relevant messages. Therefore, it may be implemented with specialized heap structures such as the interval heap [17]. Due to the limited size of the queue, binary search trees such as red-black-trees may also provide sufficient efficiency.

B. *Improvement: Avoiding of Multiple Buffered Messages from a Single Sender*

The simple priority queue may store multiple messages of the same sender. If younger messages are more relevant, they will be processed before older messages of the same sender, possibly leading to confusion in applications. Moreover, processing of older messages seems a waste of capacity if younger messages of the same sender are available in the queue. Therefore, we propose data structures and algorithms to avoid multiple buffered messages from a single sender at a time.

To implement the above presented idea, the queue needs to be checked for older messages of the same sender when a new message arrives. To avoid a time-consuming iterative checking, we propose a second data structure that uses a hash table to quickly determine whether a message of the same sender is contained in the queue. To that end, we generate a vehicle entry containing the sender ID of a message when storing the message in the queue, hash the vehicle entry into a hash table using the sender ID, and bidirectionally link the vehicle entry with the corresponding message. If the field in the hash table is already occupied by another vehicle entry, the new entry is appended in form of a double-linked list. Figure 1 illustrates that a message is queued according to its relevance and associated with a vehicle entry in the hash table.

When a new message arrives, the hash table is first checked for a vehicle entry with the same sender ID. If such an entry exists, an older message of the same sender is found. That message is removed from the queue, valuable information is extracted and appended to the new message. This valuable information may be extracted from old messages containing data fields–such as the position history–which are not part of the new message according to [5]. The old message is deleted, the new message is bidirectionally linked with the vehicle entry, and inserted into the queue according to its relevance. If there is no entry for the same sender ID, a new one is created, associated with the message, and hashed into the table when storing the message in the queue. When a message is deleted from the queue for processing, the vehicle entry is also removed from the hash table.

C. *Improvement: Relevance Decrease over Time for Buffered Messages*

So far the system uses the messages' tagged relevance for selection decisions. However, the information contained in messages loses accuracy over time as vehicles move and

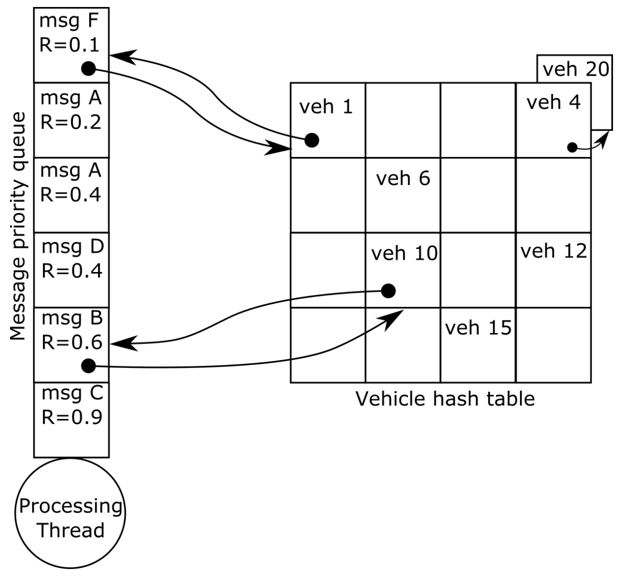


Fig. 1. Messages are linked to a vehicle entry which is hashed into a hash table. This structure helps to find old messages of the same sender with only little overhead when a new message arrives.

new messages are sent. Therefore, we want to avoid that the buffer is filled with many old relevant messages so that new, slightly less relevant messages are dropped. To that end, we want to reduce the relevance of buffered messages over time. Adapting the relevance of all buffered messages according to the presented REFs would be time-consuming. Instead of modifying their tagged relevance we rather increase the relevance of new messages by a growing offset $R_{\text{offset}}(t)$ upon arrival, which corresponds to a linear relevance decrease of buffered CAM messages.

The offset for a message at arrival time t is computed by $R_{\text{offset}}(t) = \frac{t - t_{\text{start}}}{\alpha}$. The aging parameter α is given in seconds and controls how quickly the priority of a message is reduced. As the offset increases by 1 within α time and relevance values are limited by 1, any message arriving α time later than an old message will be more relevant and may replace the old message if the buffer is fully occupied.

For reasonable values of α, the data type `double` allows $R_{\text{offset}}(t)$ to grow without arithmetic overflow for very long time.

IV. Evaluation Environment

Our evaluation environment contains two main modules: a real communication unit and a road traffic simulation. We describe them in the following. Figure 2 depicts the overall information flow in our evaluation. For the sake of completeness, the dotted parts in the communication unit illustrate the 802.11p communication stack which is not leveraged in our evaluation.

A. *Simulation Methodology*

To generate realistic CAM input for the communication unit, we apply our tool chain presented in [8].

It comprises the open source traffic simulator SUMO which is chosen due to its efficient implementation. It allows for road

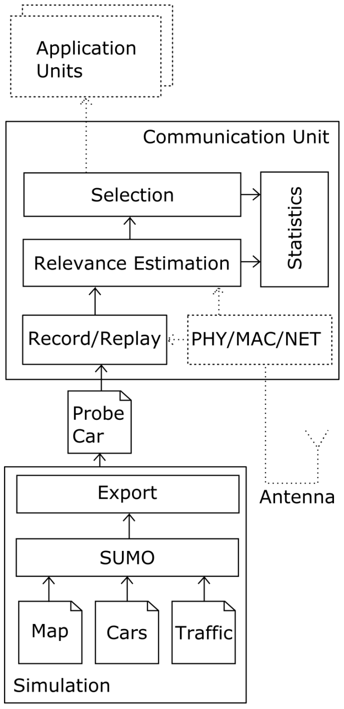


Fig. 2. Simulation data are replayed and used as input for a real communication unit that serves for runtime measurement and yields delay statistics for buffered messages.

```
90900 move 50.929612 6.951889 0 0 0 0 -13.18 13.84 3 0
90932 CAM "000000000037C8014987349873367[...]"
90972 CAM "000000000037C8014577245772363[...]"
90980 CAM "000000000037C8014767847678365[...]"
91000 move 50.929623 6.951885 0 0 0 0 -12.12 13.78 3 0
91041 CAM "000000000037C8014675346753363[...]"
91071 CAM "000000000037C8014721347213364[...]"
91091 CAM "000000000037C8014712547125364[...]"
```

Fig. 3. Excerpt of a generated replay file for a probe vehicle. It shows CAMs from three neighboring vehicles. The first column is a time stamp. Lines with `move` contain movement updates for the probe vehicle such as GPS position, speed and heading. `CAM` lines represent received CAMs from neighboring vehicles, encoded as byte string.

traffic simulation with thousands of vehicles [18]. Its inputs are road maps, vehicle and driver definitions, and vehicle routes. Based on SUMO's traffic traces we determine transmission instants of CAMs for any vehicle in the simulation according to the current movement pattern as specified in [5].

Then, we use the statistical channel model from [7] to determine which CAM is successfully received by a vehicle. Finally, we select all CAMs received by one probe vehicle and feed them into the enhanced communication unit together with its own movement in the form of a single file. Figure 3 shows an excerpt of such a file. With this methodology we analyzed over 90,000 CAMs in total. The CAMs are encoded using the standardized format with an ASN.1 library.

The effect of the message drop feature, the elimination of old messages, and the relevance decrease over time can be observed best in overload situations where the message buffer is mostly filled. Therefore, we choose for simulation a high-load scenario for which we observed very high received message rates in [8]. It consists of four parallel lanes in each driving direction on the Autobahn A5 near Frankfurt/Main, Germany. Additionally we use the *TAPASCologne* scenario as a representation for realistic traffic from the metropolitan area of Cologne, Germany [19]. This scenario contains detailed road topology data and traffic patterns defined by an open source community.

B. *Communication Unit*

The communication unit is also illustrated in Figure 2. It contains the communication stack and additional testing functions. We integrated the REFs and the selection algorithm into the stack such that every received CAM is processed by these modules.

The communication stack contains a record and replay function which is used to store data from real test drives and to replay it on desktop computers. We use that function to inject the files with the simulation data for the test vehicle. The complete communication stack, except for the lower layers, processes this replay data as it was received via the wireless channel. Additionally, we implemented a statistics module which determines the runtime of given code with an accuracy in the order of microseconds.

We run our software on the Communication Control Unit (CCU) developed for the German field test sim^{TD}. The CCU contains a single core 400 MHz Freescale PowerPC CPU with 256 MByte RAM and 512 MByte flash memory. Its core task in sim^{TD} was to handle the wireless VANET communication and the connection to the in-car networks [20].

V. Computation Time of REFs

We evaluate the computation effort for various REFs on our CCU. The results show that the implementation of proposed REFs is sufficiently efficient to be used in series control units.

We measure the average elapsed time on the evaluation platform during execution. Moreover, we use the profiling tool Valgrind/Callgrind [21] to determine the average number of instructions for the computation of relevance values.

Table II lists the average execution times in milliseconds for all REFs and also provides the execution time relative to the distance-based REF. We observe that all REFs have an average runtime around 0.05 ms to compute the relevance for a single message. An exception is the dynamicRE, but approximating dynamicRE by mappedRE yields again runtimes around 0.05 ms.

Our analysis of received message rates in [8] resulted in 500 messages per second in an overload scenario. With the measured REF runtimes of around 0.05 ms per message such rates can be well supported by the given hardware.

The results of the runtime measurement depend on the evaluation hardware and its software setup. On a multiprocessing system each process may be interrupted by the scheduler to execute other processes. Therefore, the measured runtime may include the intermittent execution of other processes. In contrast, the number of CPU instructions of an algorithm is constant for a given platform. The instruction count can be

REF	Average runtime [ms]	Relative runtime	Average number of instructions
distanceRE	0.04688	1	1621
staticRE	0.0603	1.29	2176
dynamicRE	1.0056	21.45	126,899
mappedRE	0.0633	1.35	2625
encounterRE	0.04651	0.99	1694

TABLE II
RUNTIME PERFORMANCE OF VARIOUS REFS AVERAGED OVER ABOUT 30,000 MESSAGES.

used to extrapolate the execution performance of an algorithm on other hardware of the same platform. Table II shows that the number of instructions is in a similar order of magnitude for all REFs except for dynamicRE.

VI. PERFORMANCE EVALUATION OF THE MESSAGE SELECTION MECHANISM

In this section, we evaluate relevance-specific performance metrics that illustrate the effect of the proposed CAM selection mechanism. We perform this study with staticRE as REF and normalize its return values by linear interpolation to the range $[0, 1]$. The message processing rate r_{process} is set to 100, whereas the rate of received messages is up to 500 per second [10]. Figures 4 and 5 show the results for the TAPASCologne scenario and the Autobahn A5 scenario, respectively.

A. Relevance Distribution of CAMs

The relevance distribution of CAMs depends on the road traffic scenario. The TAPASCologne scenario represents traffic with mostly moderate message rates. The Frankfurt A5 scenario was saturated in the sense that high rates of CAMs were received.

Figure 4(a) shows the relevance distribution of received CAMs for the TAPASCologne scenario. Almost 82% of the received messages have relevance values below 0.1. This figure also shows that almost all received CAMs in the TAPASCologne scenario are selected. There is only a small percentage of dropped messages (0.3%, loss probability is the difference of the values on the top and on the bottom of the bars in the figure). This means that in this scenario there is no overload most of the time.

Figure 5(a) shows the relevance distribution of received CAMs for the Frankfurt A5 scenario. Only 6.4% of the CAMs have a relevance value of more than 0.5, while 69.2% have a very low relevance of less than 0.1. Thus, only a minority of CAMs has high priority and should be preferentially treated, i.e., delivered without loss and with only little delay. The relative relevance distribution of selected CAMs shows that beginning at relevance values from 0.3 and above all messages are selected by the system. Many messages with relevance values below 0.3 are dropped due to overload.

B. Waiting Times

As the message age has impact on the usefulness of a message for applications, we measure the time messages wait in the buffer until they are selected by the system.

For the TAPASCologne scenario Figure 4(b) shows that even if almost all received CAMs are selected, their waiting times depend on each CAM's relevance value. We observe a slight increase of waiting times for relevance values of 1 down to 0.1. Highly relevant messages are selected within 10 ms, whereas messages with relevance values between 0.1 and 0.2 have a waiting time of mostly less than 50 ms. Only messages with very low relevance values below 0.1 have an average waiting of 220 ms.

In the Frankfurt A5 scenario Figure 5(b) shows that selected CAMs with a relevance between 0.5 and 1.0 face an average waiting time of less than 50 ms and a 95%-quantile of less than 75 ms. Very relevant CAMs with values above 0.9 are all processed even within 10 ms. In contrast, CAMs with a relevance below 0.1 or between 0.1 and 0.2 wait on average 1689 ms or 425 ms with 95%-quantiles of 7.5 s and 3.2 s, respectively.

Thus, highly relevant CAMs are preferentially treated. However, some little relevant CAMs are delivered so late that they are unlikely to be useful for applications, which will be improved by relevance reduction over time.

C. Impact of Relevance Aging over Time

We evaluate the relevance reduction over time by applying the aging parameters $\alpha = 10$ s and $\alpha = 1$ s to the same simulation traces.

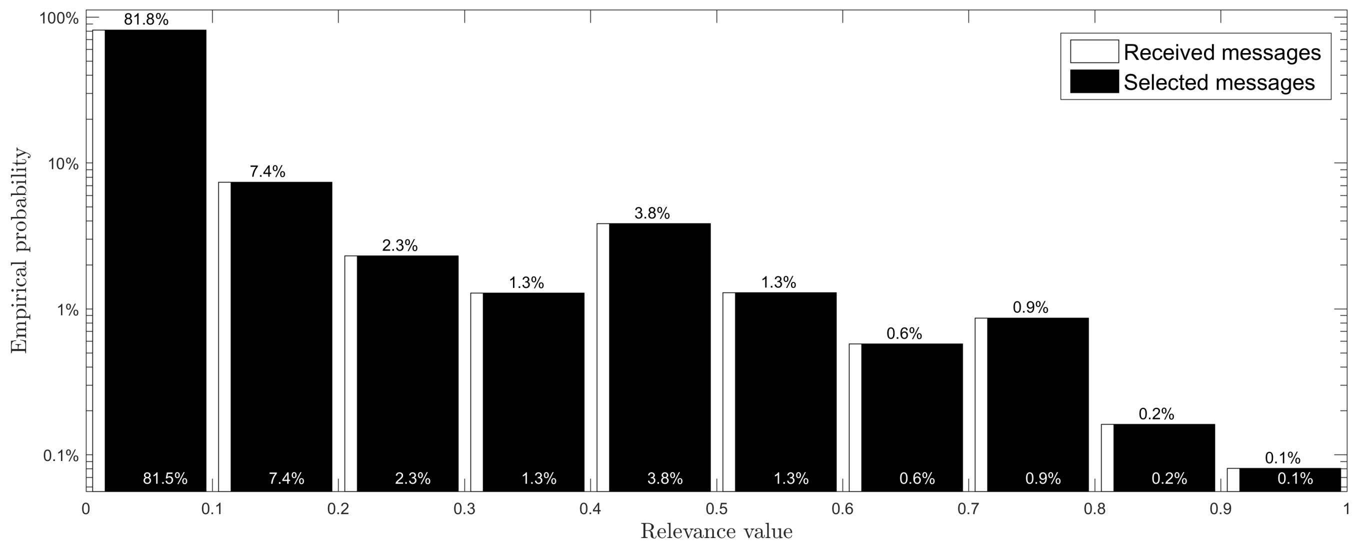
In the TAPASCologne scenario the influence of the aging parameter α is only relevant for CAMs with very little relevance. For $\alpha = 10$ s their average waiting time decreases to 140 ms.

For the Frankfurt A5 scenario the average waiting time of CAMs with low relevance decreases to average values of 341 ms and 75 ms with maximum values of 1.1 s and 80 ms. This is a significant improvement for data freshness.

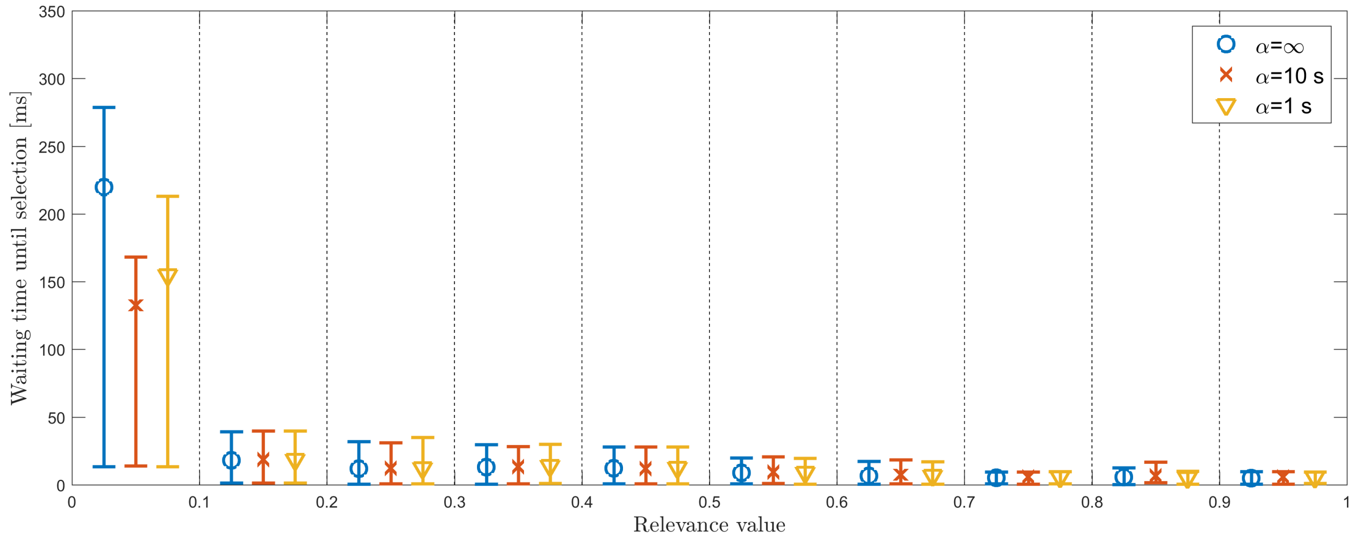
For the selected simulation data and hardware, $\alpha = 1$ s seems to be a reasonable setting. It results in waiting times below 100 ms for most messages even for medium relevance values. However, if the hardware resources are modified or other scenarios are more relevant for the applications, another parameter setup may yield better results.

VII. CONCLUSION

Vehicles may exchange large rates of Cooperative Awareness Messages (CAMs) in the near future which poses a challenge for communication control units (CCUs). They quickly need to process the most relevant CAMs and deliver them to applications. We proposed a message selection mechanism for that purpose which requires a relevance estimation function (REF) to preferably process most relevant CAMs. We



(a) Relevance distribution of received (percentage values over the bars) and selected (percentage values inside the bars) CAMs on a logarithmic scale.



(b) Relevance-dependent waiting times of selected CAMs for different aging parameters α. 5%-quantiles, means, and 95%-quantiles are given.

Fig. 4. Results for the TAPASCologne scenario.

implemented several REFs whose quality has been compared in [11] on a CCU and showed that the runtime of most of them is short enough to compute relevance values for all CAMs. The simulative performance evaluation showed that message selection drops little relevant CAMs under overload and thereby increases the average relevance of processed CAMs. Most CAMs have low relevance. Highly relevant CAMs are almost all processed with little delay. Little relevant CAMs are likely to be selected with more delay. We showed that this tradeoff can be effectively controlled by an aging mechanism which is another proposed optimization.

REFERENCES

[1] J. Harding, G. Powell, R. Yoon, J. Fikentscher, C. Doyle, D. Sade, M. Lukuc, J. Simons, and J. Wang, "Vehicle-to-Vehicle Communications: Readiness of V2V Technology for Application: DOT HS 812 014," Washington, DC, 2014.

[2] Car 2 Car Communication Consortium, "Memorandum of Understanding for OEMs within the CAR 2 CAR Communication Consortium on Deployment Strategy for Cooperative ITS in Europe," 2011.

[3] M. Sichitiu and M. Kihl, "Inter-vehicle Communication Systems: A Survey," *IEEE Communications Surveys & Tutorials*, vol. 10, no. 2, pp. 88–105, 2008.

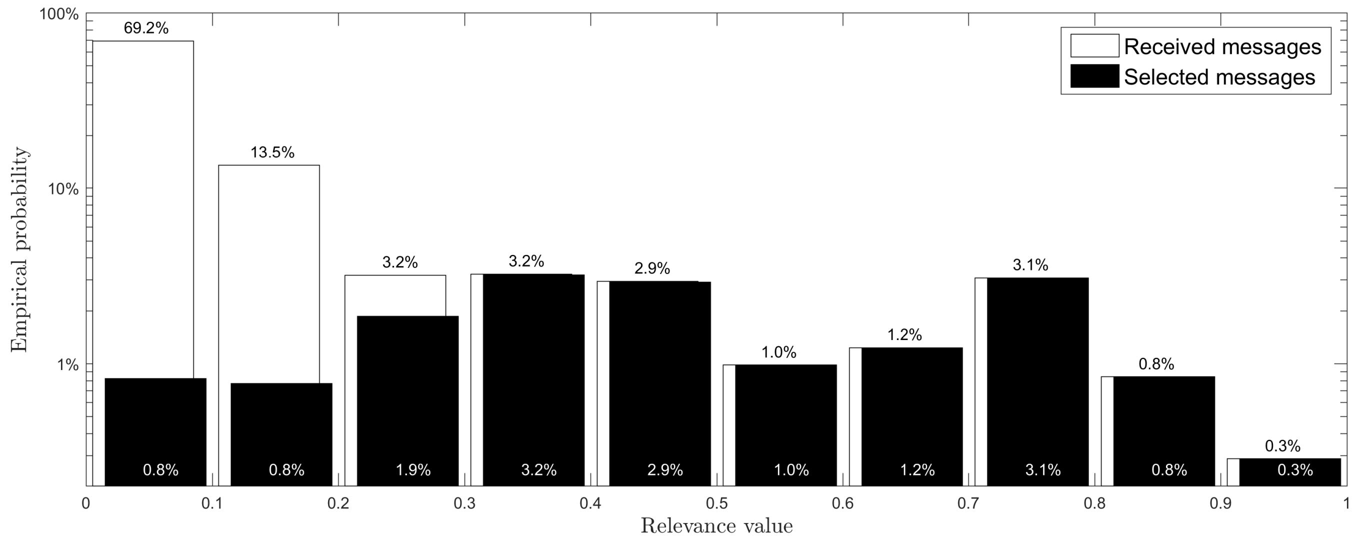
[4] D. Jiang, Q. Chen, and L. Delgrossi, "Optimal data rate selection for vehicle safety communications," in *Proceedings of the 5th ACM International Workshop on VehiculAr Inter-NETworking*. ACM, 2008, pp. 30–38.

[5] European Telecommunications Standards Institute, "Intelligent Transport Systems (ITS); Vehicular Communications; Basic Set of Applications; Part 2: Specification of Cooperative Awareness Basic Service," 2013-08.

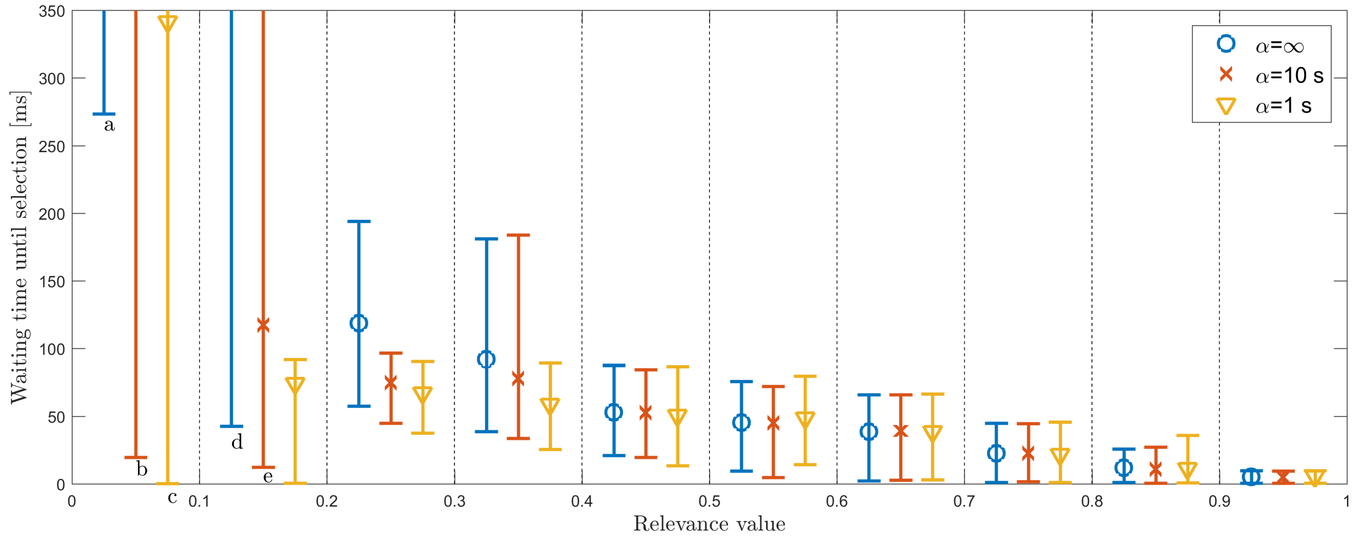
[6] N. Lyamin, A. Vinel, and M. Jonsson, "Does ETSI beaconing frequency control provide cooperative awareness?" in *IEEE International Conference on Communications Workshops*, 2015, pp. 2393–2398.

[7] J. Breu and M. Menth, "Relevance Estimation of Cooperative Awareness Messages in VANETs," *Proceedings of the 5th International*, pp. 1–5, 2013.

[8] ——, "An Improved Relevance Estimation Function for Cooperative Awareness Messages in VANETs," *Proceedings of the 6th International Workshop on Communication Technologies for Vehicles*, vol. 8435, pp. 43–56, 2014.



(a) Relevance distribution of received (percentage values over the bars) and selected (percentage values inside the bars) CAMs on a logarithmic scale.



(b) Relevance-dependent waiting times of selected CAMs for different aging parameters α. 5%-quantiles, means, and 95%-quantiles are given. The cropped lines have the following values (ms): a) 273/1689/7527; b) 20/585/1100; c) 0/341/1064; d) 43/425/3204; e) 13/117/492.

Fig. 5. Results for the Frankfurt A5 scenario.

[9] J. Breu, A. Brakemeier, and M. Menth, "Analysis of Cooperative Awareness Message rates in VANETs," in *Proceedings of the 13th International Conference on ITS Telecommunications 2013*, 2013, pp. 8–13.

[10] ——, "A quantitative study of Cooperative Awareness Messages in Production VANETs," *EURASIP Journal on Wireless Communications and Networking*, vol. 2014, no. 1, p. 98, 2014.

[11] J. Breu and M. Menth, "Comparison of Relevance Estimation Mechanisms for Cooperative Awareness Messages in VANETs," *Proceedings of the 2015 Vehicular Technology Conference Fall*, 2015.

[12] R. P. Singh and A. Gupta, "Traffic Congestion Estimation in VANETs and Its Application to Information Dissemination," *Lecture Notes in Computer Science*, vol. 6522, pp. 376–381, 2011.

[13] C. Adler, S. Eichler, T. Kosch, C. Schroth, and M. Strassberger, "Self-organized and Context-Adaptive Information Diffusion in Vehicular Ad Hoc Networks," *Proceedings of the 3rd International Symposium on Wireless Communication Systems*, pp. 307–311, 2006.

[14] N. Cenerario, T. Delot, and S. Ilarri, "A Content-Based Dissemination Protocol for VANETs: Exploiting the Encounter Probability," *IEEE Transactions on Intelligent Transportation Systems*, vol. 12, no. 3, pp. 771–782, 2011.

[15] Z. Li and C. Chigan, "On Resource-Aware Message Verification in VANETs," *Proceedings of the 2010 International Conference on Communications*, pp. 1–6, 2010.

[16] P. J. Davis, *Interpolation and Approximation*. New York: Dover Publications, 1975.

[17] M. D. Atkinson, J.-R. Sack, N. Santoro, and T. Strothotte, "Min-max Heaps and Generalized Priority Queues," *Commun. ACM*, vol. 29, no. 10, pp. 996–1000, 1986.

[18] D. Krajzewicz, J. Erdmann, M. Behrisch, and L. Bieker, "Recent development and applications of SUMO–simulation of urban mobility," *International Journal On Advances in Systems and Measurements*, vol. 5, no. 3 and 4, pp. 128–138, 2012.

[19] TAPAS Cologne Scenario. [Online]. Available: http://sumo.dlr.de/wiki/Data/Scenarios/TAPASCologne

[20] simTD Konsortium, "Deliverable D21.2, Konsolidierter Systemarchitekturentwurf," Sindelfingen, 09.10.2009.

[21] J. Weidendorfer, M. Kowarschik, and C. Trinitis, "A Tool Suite for Simulation Based Analysis of Memory Access Behavior," *Proceedings of the 2004 International Conference on Computational Science*, vol. 3038, pp. 440–447, 2004.