Content Registration in VANETs—Saving Bandwidth through Node Cooperation

Christian Wewetzer*, Björn Scheuermann†, Andreas Lübke*, Martin Mauve†
*Volkswagen Group
Driver Information Systems Research
Letter Box 1777, Wolfsburg, Germany
Email: {first.last}@volkswagen.de
†Heinrich Heine University Düsseldorf
Computer Networks Group
Universitätsstr. 1, Düsseldorf, Germany
Email: {last}@cs.uni-duesseldorf.de

***Abstract*—Search indices for distributed information are an important building block for many mobile, decentralized applications. However, the network load caused by nodes registering their information in the index can become quite significant especially in mobile networks like vehicular ad-hoc networks (VANETs), where nodes need to register periodically due to their mobility. Our goal in this paper is to encode this registration information as efficiently as possible. For this purpose, we first analytically study hash keys and Bloom filters as alternative encoding types of nodes' registration data. It turns out that in many situations, Bloom filters are more bandwidth efficient than hash keys, and that they are even more efficient when nodes cooperate. Many such forms of cooperation are conceivable; we define one specific scheme targeting metropolitan VANETs. Our accompanying simulation study shows that this first algorithm already allows for large bandwidth savings in content registration, compared to the alternative of registering hash keys.**

I. Introduction

Applications in VANETs are based on information, such as sensor data or media, that is exchanged among the VANET nodes. If nodes need to locate such information units[1] present in other nodes, an (either centralized or decentralized) index structure becomes an important building block. In such a search index, nodes register the information units they hold. In order to do so, they announce the keys of the information units and information about themselves, e. g., addresses or geographical locations. Other nodes may then use the index to perform searches and to determine which node to contact in order to obtain information of interest.

In VANETs, there are many types of information that could possibly be of interest for other nodes and therefore be stored in a search index. For example, sensor data may be useful for extended route planning. Specific types of such sensor data include friction coefficients, fuel consumption data, and weather data such as temperature measurements.

However, in VANETs frequent updates of the index may be expected, due to mobility, arriving or leaving nodes, or because of newly generated or outdated information. This index maintenance causes communication costs: when a node registers its current information units in the index, it needs to transmit data to the index. This data will typically contain the keys of all locally held information units, which may cause significant communication overhead. Keeping this communication effort low is our goal in this paper.

With our approach we address a scenario as illustrated in Figure 1. Here, the search index is hosted at the base stations that are interconnected. Each vehicle periodically registers its information units at its nearest base station. In this paper we regard the index itself as a black box and focus on the space efficient encoding of registrations that vehicles transmit in the wireless domain.

We first mathematically analyze and compare the space efficiency of two different registration encodings. Therein we assume that not the information units themselves are encoded, but well-defined descriptions of these information units (e.g. "friction coefficients, intersection Broadway and 105th Street"), so that the encoding can also be computed by vehicles not owning the information unit. In the first encoding scheme that we consider here, vehicles map each of their information unit descriptions to a hash key. A registration

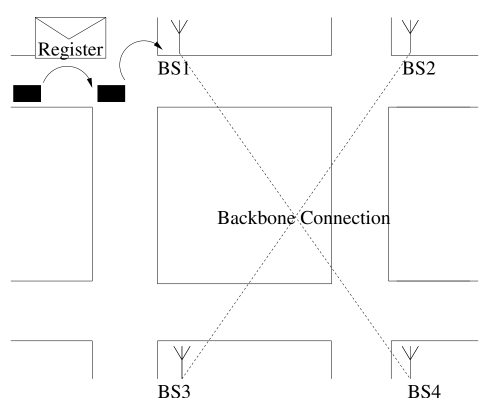


Fig. 1. Content registration in the wireless domain.

[1] Throughout this paper, we use this term to abstractly refer to one or more pieces of information, as its specific type is not important in our context.

message then consists of a list of these hash keys. In order to answer a request for some information item, the index may then simply check in which list the corresponding hash key occurred, and direct the request to the corresponding car.

As a second, alternatively viable encoding approach, we propose that vehicles insert their information unit descriptions into a container that is a Bloom filter [1]. A Bloom filter is a way to encode sets of items using a bit field. It allows to check any given item for set membership. If a registration message of a car contains a Bloom filter describing the set of information units available in that car, the index may use these Bloom filters to determine the cars to which a request should be directed.

However, care must be taken with both these encodings, because both of them bear the risk of false positives: a matching information source may be found in the index although the looked-up information unit is not registered. The probability that this happens is called the *false positive rate* (FPR). We compare the sizes of registrations in the two considered encodings under the assumption that the false positive rates of the index in total is equal. The central result of this comparison is that the Bloom filter based encoding is generally more efficient, if (lossless) compression is applied to the Bloom filters prior to their transmission.

It also turns out that the bandwidth efficiency of Bloom filters can be further improved through node cooperation. Therefore, we subsequently examine this option more closely and propose a cooperative aggregation scheme for VANETs, where vicinal nodes opportunistically build a common Bloom filter. We then conduct a simulation-based case study in a large metropolitan VANET setting, validating the feasibility and efficiency of the Bloom filter based index as well as the effectivity of our aggregation scheme.

The remainder of the paper is structured as follows: in Section II we present related work, introducing the concepts of hashing, Bloom filters, and aggregation. This is followed by an analytical comparison of the registration bandwidth of hash keys and Bloom filters in Section III. In Section IV, we present a local opportunistic aggregation scheme for Bloom filters, targeting a metropolitan VANET scenario. Using Bloom filters to reduce the registration bandwidth overhead involves some tradeoffs, which we discuss in Section V. We then verify the feasibility and efficiency of the aggregated Bloom filter based registration scheme in a large VANET scenario using simulation in Section VI, before we conclude this paper in Section VII.

II. Related Work

A. Hashing and Bloom filters

This paper considers hash keys and Bloom filters as possible encodings of nodes' information units for registration in the search index. In our work, we focus on the resulting network load in the wireless domain. We therefore do not need to consider the internal organization of the index within the network of interconnected base stations. In case of hash keys, one option would be to organize it as a distributed hash table (DHT) [2], [3].

The use of hashing in the search index bears the risk of false positive lookups due to hash collisions. If an index is based on hash keys as outlined above, assuming a perfectly random hash function with a fixed length of c bits, the probability that two different information units have the same hash key is 2^{-c}. Consequently, given that n information units are present in the index in total, the resulting FPR is

$$R_{\text{hash}}(c,n) = 1-(1-2^{-c})^n. \tag{1}$$

Alternatively to hash keys, nodes may encode their information units into Bloom filters to register in the search index. A Bloom filter [1] is a probabilistic representation of a set I of information units $\{i_1, i_2, \ldots, i_n\}$. A membership query of the Bloom filter is answered with *yes* or *no*. When the answer is *no*, the information unit queried is not part of the filter. When the answer is *yes*, it can be assumed that the information unit is part of the filter, but there is again some remaining probability of a false positive. The filter consists of a bit array $B = b_1,\ldots,b_m$. k independent hash functions $h_1(i),\ldots,h_k(i)$ are used to map an information unit i to values in the range $\{1,\ldots,m\}$. When an element i is inserted, the corresponding bits $b_{h_1(i)},\ldots,b_{h_k(i)}$ are set to 1. A membership query for i is answered with *yes* if and only if all the bits $b_{h_1(i)},\ldots,b_{h_k(i)}$ are set to 1. Given m and n, the Bloom filter yields the lowest FPR if $k = ln(2)\frac{m}{n} = ln(2)c$ [4]. Here, c again denotes the number of bits spent per information unit in the registration message. In the case of Bloom filters, c is implicitly determined by the chosen bit field size m and the number of information units n: $c = \frac{m}{n}$. The resulting FPR of a Bloom filter as a function of c is

$$R_{\text{BF}} = 2^{-\ln(2)c}. \tag{2}$$

In practice, k must of course be an integer.

Because of their space efficiency, Bloom filters have many applications in networks where bandwidth is costly. Here we focus on applications in MANETs; for a broader survey, we refer to [4]. In MANETs, Bloom filters have been suggested for discovery of services and resources [5], [6]. Presence detection [7] is another application, where nodes share an aging Bloom filter containing the IDs of active nodes in the MANET. The common element of the presented applications is that nodes proactively distribute the filter either in their close vicinity or in the whole MANET. In contrast, in our approach the filter is transmitted only to one single location, to the index.

An important question for our goal of bandwidth efficient registration is whether there exist even more space efficient data structures than the original Bloom filter. In this context, the question arises whether it is possible to give an information theoretical lower bound for the space m required by any data structure capable of representing sets of size n with a maximum false positive rate R. In fact, this lower bound is

$$m \geq n \cdot \log_2(1/R). \tag{3}$$

A derivation of this bound can be found in [4]. It turns out that the Bloom filter uses a factor of approximately 1.44 more space than a space optimal data structure with identical functionality. Mitzenmacher [8] has shown that this can be improved by compression, after which Bloom filters can theoretically achieve space optimality: the FPR of a compressed Bloom filter is

$$R_{\mathrm{CBF}} = 2^{-c}. \tag{4}$$

In this paper we consider both uncompressed and compressed Bloom filters for the search index.

Besides Bloom filters, several other space efficient or space optimal data structures have been suggested [4], [9], [10], [11], [12], but either their construction is computationally inefficient or they have other properties limiting their practical use.

Several extensions for Bloom filters have been proposed, providing Bloom filters with the ability of counting and deletion of members [13], [14], or allowing them to store values (associative array) [15]. Boldi and Vigna generalized Bloom filters, obtaining a data structure called *compact approximator* [16]. With its help, functions can be approximated in the sense that the function value returned by the compact approximator is equal or greater than the real function value. Boldi and Vigna found that this is useful in text search based on the Boyer-Moore algorithm [17].

B. In-network Data Aggregation

In-network aggregation of data, as we propose it for Bloom filters in Section IV, is an active field of research in the context of wireless sensor networks. The work of Fasolo et al. [18] surveys research results in this area. The authors classify aggregation techniques into either lossy or lossless and into duplicate sensitive or duplicate insensitive.

Our proposed local opportunistic aggregation scheme for Bloom filters in metropolitan VANETs is lossy in the sense that only the identity of the node aggregating the filters is preserved in the resulting message while the identities of the other nodes are lost. Our scheme is lossless in the sense that the aggregated Bloom filter contains the same information at the same FPR as the atomic Bloom filters. Our scheme is duplicate insensitive: the same bits are set in a common Bloom filter, no matter how many nodes register a particular information unit.

The authors of [18] further mention that the theoretical limit of bandwidth reduction through aggregation techniques depends on the correlation of data that has to be aggregated. All our calculations in this paper are based on the assumption that each node carries unique information units. Thus, the resulting savings in network load of the search index represent the worst case.

In our work, we aggregate descriptions of information units (meta information). There is other work on the aggregation of information units themselves in VANETs, see for instance [19], [20], [21], [22], [23], [24].

III. Hash Keys and Bloom Filters—Analytical Comparison of Registration Sizes

As pointed out in Section II, both hash keys and Bloom filters bear the risk of false positives, depending on their configuration. We now calculate the total size of registrations using the two data structures. Alongside, we consider the option of compressing the data structures. Afterwards, we compare the indices under the condition that the total number of information units in the network and the index FPRs are identical.

A. Total Size of Registrations

The FPR of an index that consists of a list of hash keys is given by (1). To achieve a given index FPR R, each node has to adjust the length of its hash keys. Note that hash values of all reasonable sizes can easily be constructed by using a standard hash function (e. g., MD5 [25] or SHA-1 [26]) and truncating it after the desired number of bits. For fairness reasons, we assume that all nodes are to use the same value c. Then, with n information units in the network, the total size of all hash keys in the index for a given FPR R is

$$m_{\mathrm{hashI}}(n,R) = n \cdot c = n \cdot \frac{-\ln(1-(1-R)^{\frac{1}{n}})}{\ln(2)}. \tag{5}$$

One might suggest to apply (lossless) compression to reduce the size of the list of hash keys in a registration. Unfortunately, given that good hash functions are used, the entropy of each bit in a hash key is one. Consequently, the size of the transmitted lists of hash keys can not be further reduced this way.

Let us now consider the encoding of the n information units into a total of $t \leq n$ Bloom filters with an identical number of bits per information unit c, i. e., with the same total size of the registration messages. Then, checking an index of t Bloom filters, each with a FPR as given by (2), yields

$$R_{\mathrm{BFI}}(c,t) = 1-(1-2^{-\ln(2)c})^{t}. \tag{6}$$

Therefore, for a given FPR R and a total of n information units in the index, the total size of t optimally constructed Bloom filters is

$$m_{\mathrm{BFI}}(n,R,t) = n \cdot c = n \cdot \frac{-\ln(1-(1-R)^{\frac{1}{t}})}{\ln(2)^2}. \tag{7}$$

If the index is formed by t compressed Bloom filters, each with FPR as given by (4), its FPR is

$$R_{\mathrm{CBFI}}(c,t) = 1-(1-2^{-c})^{t}. \tag{8}$$

The total size of the t filters for a given index FPR R is then

$$m_{\mathrm{CBFI}}(n,R,t) = n \cdot \frac{-\ln(1-(1-R)^{\frac{1}{t}})}{\ln(2)}. \tag{9}$$

B. Comparison

We now compare the sizes of the registration data in the different index types under the condition that n and R are equivalent.

Comparing the total registration size of an index of standard Bloom filters given by (7) to the total registration size of an equivalent index of compressed Bloom filters given by (9), the total size of compressed Bloom filter based registrations is only a factor of $\ln(2) \approx 0.693$ of that of standard Bloom filter based registrations.

Comparing the registration sizes of an index of compressed Bloom filters and an equivalent index of hash keys (given by (5)) yields the highly interesting result that *the more information units can be aggregated into one filter (and thus the smaller the total number t of filters), the greater are the bandwidth savings compared to a hash key based index*. As soon as there is one filter containing more than one information unit (yielding $t < n$), the total required bandwidth of the compressed Bloom filters is lower than that of the hash table. This can already be achieved without any inter-node cooperation, if nodes carry more than one information unit. The situation becomes even better if nodes cooperatively form larger compressed Bloom filters, containing descriptions of the information units from whole groups of nodes. We detail one algorithm for such aggregated registration in Section IV.

The outcome of the comparison between total registration sizes of *standard* (i. e., uncompressed) Bloom filters and the hash key based index is not that obvious. Setting $m_{\mathrm{BFI}} < m_{\mathrm{hashI}}$ gives

$$t < \frac{\ln(1-R)}{\ln(1-(1-(1-R)^{1/n})^{\ln(2)})}. \qquad (10)$$

This means that the total size of the standard Bloom filters is below that of the hash keys only if a certain level of aggregation can be achieved, depending on R and n. Figure 2 depicts the value of t for which the total registration sizes of hash key based index and Bloom filter index are equal, dependent on R and n (please note the logarithmic axis of R). With decreasing FPR, more aggregation is required to keep the total size of Bloom filters below that of hash keys. On the other hand, the more information units there are in the network, the more filters may be created.

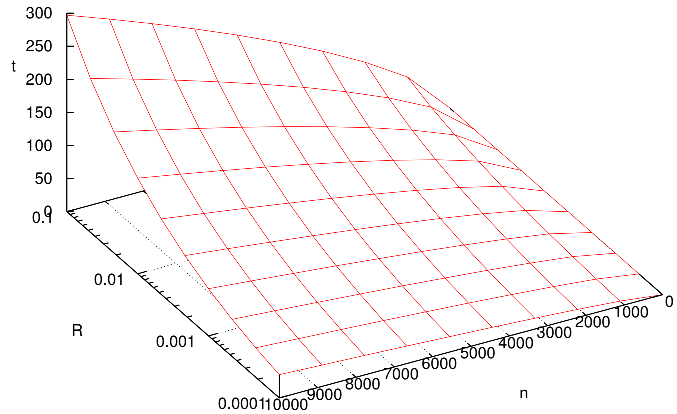


Fig. 2. Total number t of Bloom filters in the network when their total registration size is equal to that of a hash key based index, depending on number of information units n and FPR R.

IV. Cooperative Aggregation of Bloom Filters in VANETs

In a city, often clusters of vehicles emerge, for example when vehicles wait at intersections because of cross-traffic or traffic lights. This observation is the base of our opportunistic aggregation scheme. For this scheme, we define two time periods: a minimum time span (t_{min}) and a maximum time span (t_{max}) between two consecutive registration messages transmitted by the same vehicle. Vehicles *may* register their keys after the minimum time has elapsed, they *must* register after waiting for the maximum time. Whenever a vehicle must register its keys, it becomes a local aggregation coordinator. The coordinator and all vehicles in its radio range that *may* register their keys cooperatively create one common Bloom filter.

The process of creating this common Bloom filter works as follows: the coordinator determines the total number of information units to be registered, based on the number of its own information units and the corresponding numbers of its neighbors that may register. The latter information is part of the periodic vehicle beacon messages. Based on this number, the coordinator computes the size m of the combined Bloom filter. The coordinator broadcasts m, whereafter all its neighbors that *may* register keys reply with a Bloom filter of this size, containing their registration information. We call such a Bloom filter an *atomic Bloom filter*. The coordinator combines the atomic Bloom filters using a bit-wise OR operation, which yields the *combined Bloom filter*. It then registers the combined Bloom filter in the search index.

With aggregation, only the coordinator can be identified as the originator of the Bloom filter; the original information sources cannot be directly identified when the index is queried for their keys. To solve this problem, we propose a two-step forwarding of requests: first, the incoming request is forwarded to the coordinator of the Bloom filter which contains the requested data. Second, when the coordinator receives the request, it forwards it to the vehicle whose Bloom filter contained the requested key. For this it is necessary that the coordinator caches all messages sent by its neighbors when creating a common Bloom filter. To avoid the coordinator becoming a single point of failure, all vehicles listen for atomic Bloom filters and forward incoming requests in case they overheard the registration of the information source.

V. Trade-offs in Bloom Filter based Index Queries

We have seen that by using Bloom filters, content can be registered more bandwidth efficiently than through hash keys. Nevertheless, it should be noted that the network load of querying a Bloom filter based index is higher than that of querying an index of hash keys: for membership checks of the Bloom filters in the index, it is required to transmit the

k hash values of the information unit description. Sending a single hash value (as in a hash key based index query) is not sufficient.

As an apparent alternative, the description of the information unit itself could be sent instead of its associated encoding, allowing index nodes to compute the k hash functions, but potentially causing even higher network load if the descriptions are long.

As another idea, there is a way to query Bloom filters with k hash functions by only sending two hash keys. This requires a slightly different construction of the filters: Kirsch and Mitzenmacher in [27] showed that the k hash values $h_0, \ldots, h_{k-1}$ for Bloom filters can be obtained through only two (independent and fully random) hash functions $g_1(x)$ and $g_2(x)$:

$$h_j(x) = (g_1(x) + j g_2(x)) \bmod m \text{ for } j = 0, \ldots, k-1 \quad (11)$$

A Bloom filter constructed in this way has the same asymptotic FPR as the standard Bloom filter. For checking the index, it thus suffices to transmit $g_1(x)$ and $g_2(x)$.

VI. A Case Study: Search in a Sparse Metropolitan VANET

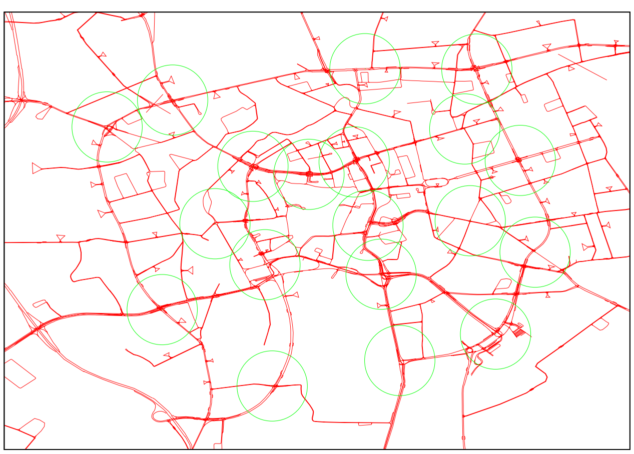


Fig. 3. Sample scenario, $(4 \times 4\,\text{km})$ size.

In this section, we investigate index based information search in a VANET. The considered scenario (Figure 3, loops and dead ends are inflows and outflows of vehicles in the simulation) is a mixed wired-wireless scenario consisting of the VANET as well as interconnected base stations, located at the center of the circles in the figure. It is modeled after a real German city. The search index is hosted at these base stations. Figure 4 depicts the total number of vehicles present in the simulation area during simulation time. The *penetration ratio* is defined as the fraction of all vehicles in the scenario that is equipped with VANET technology. With 10 % (20 %) penetration ratio, about 200 (400) vehicles are present.

In the following, we first outline the simulation setup (Section VI-A). Thereafter, we present the results (Section VI-B), answering two key questions: first, whether the search success rate using an index of aggregated Bloom filters is equivalent to that of querying one that is based on lists of hash keys; second, we determine the efficiency of the proposed aggregation scheme. Determining the achievable level of aggregation allows us to calculate the bandwidth usage depending on the number n of information units in the network and the intended FPR R for all considered registration encodings (Section VI-C).

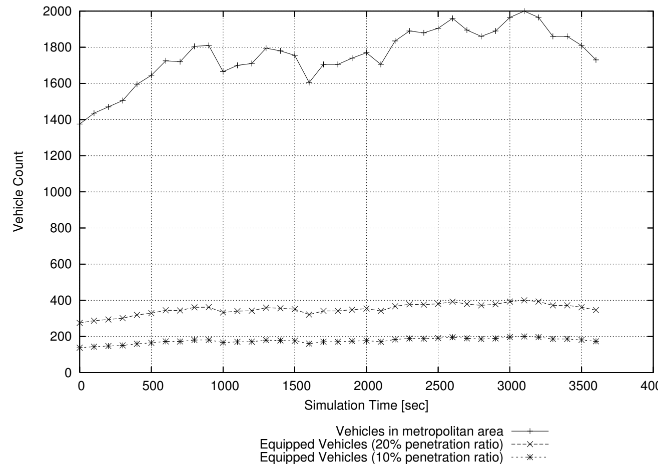


Fig. 4. Sample scenario, $(4 \times 4\,\text{km})$ size.

A. Simulation Setup

To evaluate the search indices, we used a combined simulation environment consisting of the traffic simulator VISSIM and the network simulator ns-2 [28]. With help of this simulator coupling, we may simulate scenarios containing several thousands of vehicles. VISSIM applies the psychophysical driver behavior model developed by Wiedemann [29], which results in highly realistic vehicular movement patterns. The vehiclar traffic pattern has been calibrated based on real-world measurements in the modeled city. The simulation environment is enhanced with a worst-case model of the effect of buildings on radio transmission: buildings are modelled as polygons in the simulation scenario. Radio transmission between two vehicles is impossible if their line of sight intersects such a polygon.

Table I contains the parameters of the simulation. The values for the beacon interval and the registration interval were tuned for high search performance: lower values do not significantly improve search success rate. In hash-based registration, vehicles register their information units in time interval t_{max}. In the cooperative Bloom filter-based scheme, vehicles register in the time span between t_{min} and t_{max} (cf. Section IV).

To evaluate the search performance, we restricted new search requests to information available in the VANET. This means that a suitable information source is present at the time the request is created; it does imply that the corresponding registration is available in the index. We here use a replication

TABLE I
SIMULATION PARAMETERS.

Wireless Network Type	IEEE 802.11b
Channel Model	two-ray ground
Radio Range	225 m
Penetration Ratio	10 %,20 %
Simulation Duration	3600 s
Information Count per Vehicle	100
Minimum Request Interval per Vehicle (t_{min})	75 s
Maximum Request Interval per Vehicle (t_{max})	100 s
Replication Factor	1
Local Broadcast Radius	200 m
Beacon Interval $t_{BeaconInterval}$	2 s
Register Interval $t_{repeatRegister}$	30 s

factor of 1, meaning that every information is only once available in the VANET.

Regarding the information count per vehicle, a realistic number can hardly be predicted because it depends on the applications using the search index. In [30] and [31] the numbers of files per host in Internet P2P file sharing systems has been investigated. Note that these systems are mainly used for sharing of large audio and video files, which can not be assumed to be the main type of contents in a VANET, due to bandwidth restrictions. However, we used the results on Gnutella and Napster from [30] as a rough guideline and chose an information count of $n = 100$ in the following evaluation. Consequently, each node registered 100 information units.

B. Simulation Results

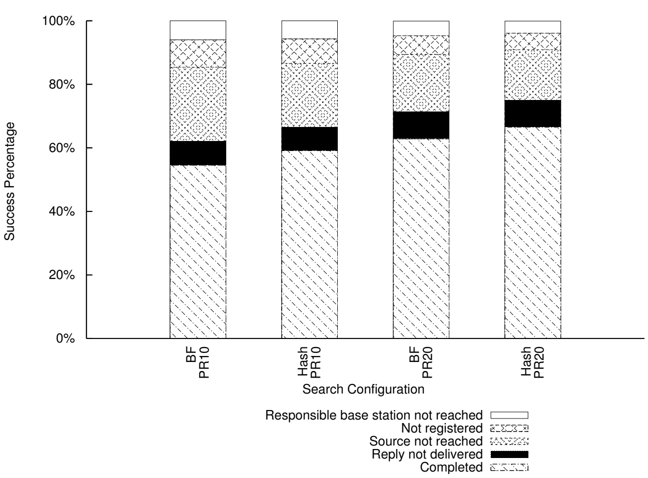


Fig. 5. Search success rates using Bloom filter based index (BF) and hash key based index (Hash) at penetration ratios 10% (PR10) and 20% (PR20).

1) Comparison of Success Rates: Figure 5 depicts the success rates of search requests based on the Bloom filter based and hash key based index. The hash key based index has an about 2 % better success rate than the Bloom filter based index. The slightly worse performance of the Bloom filter based index is a result of the two-step forwarding process of requests that is applied because of aggregated registration: sometimes, neither the coordinator nor a node that overheard the coordination process can be contacted and thus the information source cannot be identified. With increasing penetration ratio, the consistency of both indices improves, thereby increasing the number of successfully forwarded and answered search requests.

A deeper analysis of failed searches reveals that in most cases the information source can not be reached. This happens for two reasons. First, registration information is stored in the index for a certain time; when an information source has left the scenario its information is still registered, until it times out. Thus, incoming requests are forwarded but never reach the source. Second, even if the information source is present, it may be unreachable for a longer period of time because of the partitioned VANET. When the information source moves far away from the position returned by the index, it will not receive the request.

The communication delay is also the cause of the fraction of requests failing because the corresponding registration is missing: the registration message sent by the corresponding information source has not yet arrived at a base station and is therefore not included in the index.

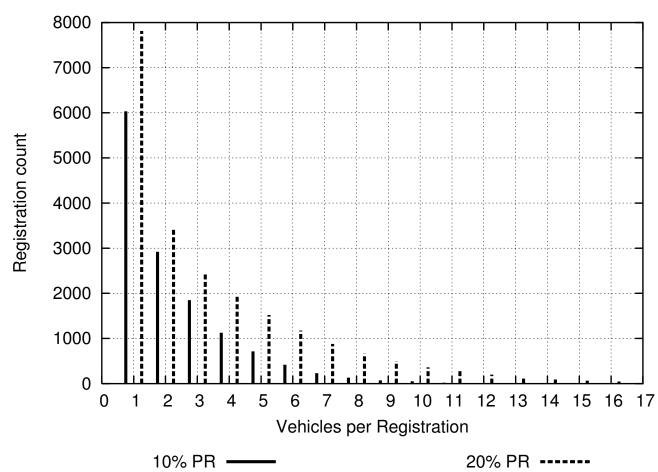


Fig. 6. Registrations per filter.

2) Efficiency of Aggregation Scheme: Figure 6 shows the effect of our proposed aggregation scheme. While most filters in the index contain the registration of only one information source, the average number of information sources per filter is 2.4 (3.6) at 10 % (20 %) penetration ratio. Thus, without aggregation, the index would contain 2.4 (3.6) times more filters. Naturally, the effect of aggregation can be expected to become more and more significant with higher penetration ratio since the number of close-by equipped vehicles increases.

C. Calculation of Bandwidth Savings

In the previous section we have determined the aggregation capabilities of the proposed algorithm in the sample scenario. At 10 % penetration ratio, the registration data of the approximately 200 nodes is bundled into $t \approx \frac{200}{2.4}$ filters. At 20 %

penetration ratio, aggregation allows to bundle the registration data of the approximately 400 nodes into $t \approx \frac{400}{3.6}$ filters. These results enable us to calculate the number of bits per information c that nodes have to use when a certain FPR of the index has to be achieved, using (5), (7), and (9), respectively.

The results are depicted in Figure 7. Note that the value c for Bloom filters is not affected by the number of information units contained in the Bloom filters, but only by the total number of Bloom filters. At both considered penetration ratios, the achievable bandwidth savings of Bloom filters are evident. For example, with 100 information units per node at 10 % penetration ratio, hash keys of length $c \approx 22$ are required to achieve an FPR of $R = 0.01$ in the index. With compressed Bloom filters, the same FPR is achieved with only $c \approx 14$ bits per information. If compression is not an option, using standard Bloom filters also allows for a lower value of c ($c \approx 19$) than an index of hash keys. If each node possesses 1000 information units, the relative savings are even higher.

It has to be noted that we have idealized the savings of the compressed Bloom filter in our calculations: the theoretically achievable FPR (given by (4)) is reached when the number of used hash functions k approaches zero and the length of the Bloom filter before compression goes to infinity [8]. To achieve the best results in practice, one would choose $k = 1$ or $k = 2$ and a large bit array that is then compressed. The resulting savings are between those of the standard Bloom filter and the idealized compressed Bloom filter [8].

D. Evaluation Summary

Our simulative evaluation of the aggregated Bloom filter based index in the sample scenario has shown that its search success rate is comparable to that of search in a hash key based index. The proposed aggregation scheme allows for an average of 2.4 (3.6) nodes combining their registration data at 10 % (20 %) penetration ratio. Based on these results, our calculation indicates that—compared to a hash key based index—bandwidth savings in the order of several ten percent are achievable with the Bloom filter based index in this scenario.

VII. Conclusion

In this paper, we have proposed an index of Bloom filters as a registration bandwidth efficient alternative to registering hash keys. We analytically confirmed the superior space efficiency of (standard and compressed) Bloom filters compared to hash keys, and in this context underlined that this advantage can be further increased through node cooperation. Besides calculating the bandwidth savings in registrations, we identified the tradeoffs in querying that occur when a Bloom filter based index is used, and we outlined possible solutions.

Building on our theoretical findings, we defined a local opportunistic aggregation scheme targeting metropolitan VANETs. We furthermore considered a sample scenario and verified that the proposed aggregated registration and search through Bloom filters gives a success rate comparable to that of search in an index of hash keys. Subsequently, we determined the level of aggregation that can be achieved through our proposed scheme. This allowed us to calculate the bandwidth savings of Bloom filters for registration in the sample scenario, depending on the intended false positive rate of the search index. Under the examined conditions, the savings are in the order of several ten percent compared to a hash table.

We conclude that in networks where registration bandwidth is critical and the identified tradeoffs in querying the index are acceptable, our proposed cooperative Bloom filter based registration presents a viable alternative to registering hash keys.

VIII. Future Work

Our proposed aggregation scheme is local and time-based. There is ample opportunity for further enhancements and variations of this scheme. We now briefly outline first ideas.

Currently, only the nodes in single-hop distance from the coordinator aggregate their registration data. Alternatively, nodes within a larger area could aggregate their filters, so that the total number of registrations in the index is even more reduced.

In the current version of the aggregation scheme, the coordination process is started when a timer expires. It is also conceivable that a node detecting a high number of to-be-registered information units in its neighborhood triggers the aggregation process, with the idea of exploiting situations in which many registrations can be combined.

Lastly, we evaluated the aggregation scheme in a city scenario under low penetration ratios. It would certainly be of interest to study its performance in other metropolitan areas and traffic situations.

References

[1] B. H. Bloom, "Space/time trade-offs in hash coding with allowable errors," *Commun. ACM*, vol. 13, no. 7, pp. 422–426, 1970.

[2] H. Balakrishnan, M. F. Kaashoek, D. Karger, R. Morris, and I. Stoica, "Looking up data in p2p systems," *Commun. ACM*, vol. 46, no. 2, pp. 43–48, 2003.

[3] C. Wewetzer, M. Caliskan, A. Luebke, and M. Mauve, "The Feasibility of a Search Engine for Metropolitan Vehicular Ad-Hoc Networks," in *AutoNet 2007: Proceedings of the 2nd IEEE Workshop on Automotive Networking and Application*, Nov. 2007.

[4] A. Z. Broder and M. Mitzenmacher, "Survey: Network applications of bloom filters: A survey," *Internet Mathematics*, vol. 1, no. 4, 2003.

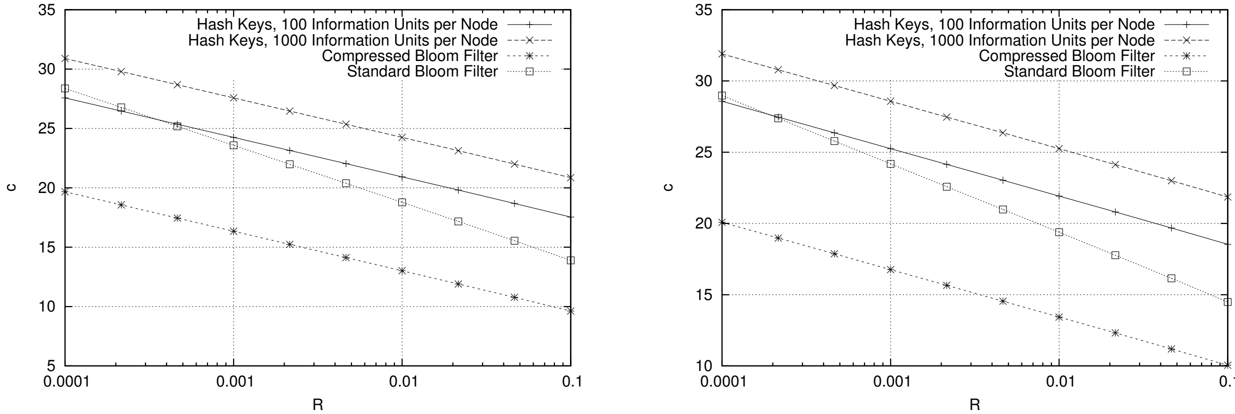
[5] F. Sailhan and V. Issarny, "Scalable service discovery for manet," in *PERCOM '05: Proceedings of the Third IEEE International Conference on Pervasive Computing and Communications*. Washington, DC, USA: IEEE Computer Society, 2005, pp. 235–244.

[6] F. Liu and G. J. Heijenk, "Context discovery using attenuated bloom filters in ad-hoc networks," *Journal of Internet Engineering*, vol. 1, no. 1, pp. 49–58, 2007.

[7] T. M. C. Tran, B. Scheuermann, and M. Mauve, "Detecting the presence of nodes in manets," in *CHANTS '07: Proceedings of the second workshop on Challenged networks CHANTS*. New York, NY, USA: ACM, 2007, pp. 43–50.

[8] M. Mitzenmacher, "Compressed bloom filters," in *PODC '01: Proceedings of the twentieth annual ACM symposium on Principles of distributed computing*. New York, NY, USA: ACM, 2001, pp. 144–150.

[9] D. Fotakis, R. Pagh, P. Sanders, and P. G. Spirakis, "Space efficient hash tables with worst case constant access time," *Theory Comput. Syst.*, vol. 38, no. 2, pp. 229–248, 2005.



(a) 10 % penetration ratio

(b) 20 % penetration ratio

Fig. 7. Required number of bits per information c to achieve FPR R in the index for hash keys, standard Bloom filters and compressed Bloom filters.

[10] P. Wolper and D. Leroy, "Reliable hashing without collosion detection," in *CAV '93: Proceedings of the 5th International Conference on Computer Aided Verification*. London, UK: Springer-Verlag, 1993, pp. 59–70.

[11] M. Dietzfelbinger and R. Pagh, "Succinct data structures for retrieval and approximate membership (extended abstract)," in *ICALP '08: Proceedings of the 35th international colloquium on Automata, Languages and Programming, Part I*. Berlin, Heidelberg: Springer-Verlag, 2008, pp. 385–396.

[12] A. Pagh, R. Pagh, and S. S. Rao, "An optimal bloom filter replacement," in *SODA '05: Proceedings of the sixteenth annual ACM-SIAM symposium on Discrete algorithms*. Philadelphia, PA, USA: Society for Industrial and Applied Mathematics, 2005, pp. 823–829.

[13] L. Fan, P. Cao, J. M. Almeida, and A. Z. Broder, "Summary cache: a scalable wide-area web cache sharing protocol," *IEEE/ACM Trans. Netw.*, vol. 8, no. 3, pp. 281–293, 2000.

[14] S. Cohen and Y. Matias, "Spectral bloom filters," in *SIGMOD '03: Proceedings of the 2003 ACM SIGMOD international conference on Management of data*. New York, NY, USA: ACM, 2003, pp. 241–252.

[15] B. Chazelle, J. Kilian, R. Rubinfeld, A. Tal, and O. Boy, "The bloomier filter: an efficient data structure for static support lookup tables," in *In Proceedings of the Fifteenth Annual ACM-SIAM Symposium on Discrete Algorithms (SODA*, 2004, pp. 30–39.

[16] P. Boldi and S. Vigna, "Mutable strings in java: design, implementation and lightweight text-search algorithms," *Sci. Comput. Program.*, vol. 54, no. 1, pp. 3–23, 2005.

[17] R. S. Boyer and J. S. Moore, "A fast string searching algorithm," *Communications of the ACM*, vol. 20, no. 10, pp. 62–72, 1977.

[18] E. Fasolo, M. Rossi, J. Widmer, and M. Zorzi, "In-network aggregation techniques for wireless sensor networks: A survey," *IEEE Communication Magazine*, Apr. 2007. [Online]. Available: http://icapeople.epfl.ch/widmer/files/Fasolo2007Survey.pdf

[19] T. Nadeem, S. Dashtinezhad, C. Liao, and L. Iftode, "TrafficView: traffic data dissemination using car-to-car communication," *ACM SIGMOBILE Mobile Computing and Communications Review*, vol. 8, no. 3, pp. 6–19, July 2004.

[20] M. Caliskan, D. Graupner, and M. Mauve, "Decentralized Discovery of Free Parking Places," in *Proceedings of the Third ACM International Workshop on Vehicular Ad Hoc Networks (VANET 2006)*, September 2006, pp. 30–39.

[21] S. Dornbush and A. Joshi, "StreetSmart Traffic: Discovering and disseminating automobile congestion using VANET's," in *VTC Spring 2007*, pp. 11–15.

[22] K. Ibrahim and M. C. Weigle, "Accurate data aggregation for VANETs," in *VANET '07: Fourth ACM International Workshop on Vehicular Ad-Hoc Networks*, 2007, pp. 71–72.

[23] C. Lochert, B. Scheuermann, and M. Mauve, "Probabilistic aggregation for data dissemination in VANETs," in *VANET '07: Proceedings of the fourth ACM international workshop on Vehicular ad hoc networks*. New York, NY, USA: ACM, 2007, pp. 1–8.

[24] C. Lochert, B. Scheuermann, C. Wewetzer, A. Luebke, and M. Mauve, "Data aggregation and roadside unit placement for a VANET traffic information system," in *VANET '08: Fifth ACM International Workshop on VehiculAr Inter-NETworking*, 2008.

[25] R. Rivest, "The MD5 Message-Digest Algorithm," RFC 1321 (Informational), Apr. 1992. [Online]. Available: http://www.ietf.org/rfc/rfc1321.txt

[26] D. Eastlake 3rd and P. Jones, "US Secure Hash Algorithm 1 (SHA1)," RFC 3174 (Informational), Sept. 2001, updated by RFC 4634. [Online]. Available: http://www.ietf.org/rfc/rfc3174.txt

[27] A. Kirsch and M. Mitzenmacher, "Less hashing, same performance: Building a better bloom filter," *Random Structures & Algorithms*, vol. 33, no. 2, pp. 187–218, 2008.

[28] C. Lochert, M. Caliskan, B. Scheuermann, A. Barthels, A. Cervantes, and M. Mauve, "Multiple Simulator Interlinking Environment for Inter Vehicle Communication," in *VANET 2005: Proceedings of the Second ACM International Workshop on Vehicular Ad Hoc Networks*, Sept. 2005, pp. 87–88.

[29] R. Wiedemann, "Simulation des Straßenverkehrsflusses (in German language)," *Schriftenreihe des Instituts für Verkehrswesen der Universität Karlsruhe*, vol. Heft 8, 1974.

[30] S. Saroiu, P. K. Gummadi, and S. D. Gribble, "A measurement study of peer-to-peer file sharing systems," in *Proceedings of Multimedia Computing and Networking 2002 (MMCN '02)*, San Jose, CA, USA, January 2002. [Online]. Available: citeseer.nj.nec.com/saroiu02measurement.html

[31] W. Wang, H. Chang, A. Zeitoun, and S. Jamin, "Characterizing guarded hosts in peer-to-peer file sharing systems," in *In Proceedings of IEEE Global Communications Conference*, ser. Global Internet and Next Generation Networks, 2004.