Probabilistic Aggregation for Data Dissemination in VANETs

Christian Lochert Björn Scheuermann Martin Mauve

Computer Networks Research Group
Heinrich Heine University
Düsseldorf, Germany

{lochert,scheuermann,mauve}@cs.uni-duesseldorf.de

ABSTRACT

We propose an algorithm for the hierarchical aggregation of observations in dissemination-based, distributed traffic information systems. Instead of carrying specific values (e. g., the number of free parking places in a given area), our aggregates contain a modified Flajolet-Martin sketch as a probabilistic approximation. The main advantage of this approach is that the aggregates are duplicate insensitive. This overcomes two central problems of existing aggregation schemes for VANET applications. First, when multiple aggregates of observations for the same area are available, it is possible to combine them into an aggregate containing all information from the original aggregates. This is fundamentally different from existing approaches where typically one of the aggregates is selected for further use while the rest is discarded. Second, any observation or aggregate can be included into higher level aggregates, regardless if it has already been previously—directly or indirectly—added. As a result of those characteristics the quality of the aggregates is high, while their construction is very flexible. We demonstrate these traits of our approach by a simulation study.

Categories and Subject Descriptors

H.4.3 [**Information Systems Applications**]: Communications Applications; C.2.4 [**Computer-Communication Networks**]: Distributed Systems; C.2.1 [**Computer-Communication Networks**]: Network Architecture and Design

General Terms

Algorithms, Design, Performance

Keywords

VANET, Aggregation, Sketch, Information Dissemination

1. INTRODUCTION

Cooperative information gathering and sharing forms a prominent class of VANET applications. For instance, approaches that disseminate traffic information like TrafficView [10] or SOTIS [17, 18] as well as a system to exchange information on free parking places [2] have been designed. These applications have in common that they distribute measurement results obtained by the participating cars in a comparatively large area.

Typically, this is accomplished in the following way. Each car makes observations. An observation is essentially some measured value (traffic density, free parking places,...), related to a position in space (i. e., a road segment or a small area) and a point in time when the observation has been made. All or part of the locally stored information is periodically broadcasted. Upon reception of such a broadcast, a node incorporates the received data into the local knowledge base. By comparing the timestamps of observations, it can ensure that always the most up-to-date value for each position is stored and redistributed. However, if we assume that the spatial density of points for which observations are made is approximately constant, the amount of data increases quadratically with the covered radius. Thus, the amount of data to be broadcasted by each car will likewise increase quickly. Since the network's capacity is constant, this is fatal for the scalability of such a system.

To overcome this problem, the use of hierarchical data aggregation has been proposed: with increasing distance, observations concerning larger and larger areas (or road segment lengths) are combined into one single value. Such an aggregated value could, for example, be the average speed on a longer road segment, or the percentage of free parking places in a part of a city. Coarse aggregates are made available at greater distances, more detailed data is kept only in the near vicinity. However, even though the idea has often been mentioned, central problems of such an approach have, so far, remained unsolved.

A fundamental issue that arises is that aggregates can not, like single observations, be directly compared regarding the up-to-dateness and completeness of the contained data. They are created by cars that will typically not have the most up-to-date measurements for all underlying points available. Therefore, multiple aggregates for the same area may exist, based on different, but likely overlapping knowledge. To decide which one is based on "better" underlying data is hard, if not impossible.

In this paper we propose an algorithm that solves this issue. We achieve this by a special data representation: both single observations and aggregates in our scheme do not carry the value of, e. g., the number of free parking places directly, but instead contain an approximation of it in form of a modified Flajolet-Martin

sketch [6]. This still does not provide a way to compare the quality of two aggregates directly, but it allows for something even better: in our scheme, multiple aggregates for the same area can be merged, yielding a new one that incorporates all the information contained in any one of the aggregates. This is fundamentally different from existing approaches where two aggregates describing the same area cannot be merged[1]. In our scheme there is no longer a need to decide which aggregate contained more up-to-date information since the resulting aggregate comprises all the information from all aggregates that have been merged.

Our approach also allows observations or lower-level aggregates to be integrated into an already existing higher-level aggregate at any time. Note that this is is not feasible with existing approaches, because it cannot be determined which data is already present in the aggregate and interesting aggregates like sums or averages are typically duplicate sensitive.

Apart from making decisions regarding the aggregate quality unnecessary, the proposed scheme also largely eases the generation of good aggregates. In order to create a sensible aggregate, a node would usually have to collect data on a significant fraction of the covered area before an aggregate that likely constitutes a good representation can be formed. With our scheme, the aggregate can instead be maintained while being passed around in the network, always incorporating new information on-the-fly. This results in aggregates of a much higher quality.

The remainder of this paper is structured as follows. In the next section we review aggregation as it has been proposed for VANET applications, as well as some previous uses of Flajolet-Martin sketches in the networking area. Thereafter, we introduce our algorithm in detail in Section 3. In Section 4, we present and discuss the results of a simulative evaluation of the algorithm in a VANET city scenario. Finally, we conclude this paper with a summary in Section 5.

2. RELATED WORK

Recently, many convenience applications for VANETs have been discussed, and often they use some form of data dissemination. While discussing them all in detail would definitely exceed the scope of this paper, we concentrate on three examples, and show how they deal with the problems considered in this study.

In the Self-Organizing Traffic Information System (SOTIS) [17, 18], information on the traffic situation is distributed opportunistically, by sending periodic beacons containing the knowledge of the sending node on the traffic situation in a larger surrounding. The authors also outline a (non-hierarchical) aggregation scheme, combining all the known information on each fixed-length road segment to one average value. Upon reception, a node considers an aggregate "better" if it has a newer timestamp. But since these timestamps are assigned when the aggregate is computed, this system exhibits the problems outlined in the introduction: a newly computed aggregate with a new timestamp is not necessarily based on the most up-to-date information, and aggregates representing largely disjoint knowledge can neither be identified as such nor can they be merged. For these reasons, a system like SOTIS could largely benefit from the aggregation scheme introduced here.

[1]Any hierarchical aggregation scheme will provide a way to merge lower level aggregates into a higher level aggregate. This is not what we refer to. The point here is that two aggregates describing the same area are merged.

TrafficView [10] is another system for disseminating traffic information, similar to SOTIS in both aims and mechanisms. The authors of TrafficView also introduce a data aggregation scheme. Different from SOTIS, TrafficView distributes information on position and speed of single vehicles. The aggregation mechanism combines a number of "similar" vehicles in an adaptive way, aiming to minimize the introduced errors. Again, each aggregate is assigned a timestamp: TrafficView uses the minimum information generation time of the combined measurements. Consequently, similar problems as for SOTIS arise. A definite decision about the relative up-to-dateness of stored and received values can not reliably be made. Again, this could be overcome by using our probabilistic aggregation scheme.

A different application of data dissemination with aggregation in VANETs is proposed in [2]: the distribution of information on free parking places. The information on the current occupancy level is generated by local infrastructure at the parking lots, and, like in the traffic information systems above, disseminated by periodic broadcasting. Aggregation is performed hierarchically in a quadtree structure over the city area. Where SOTIS uses the time of generation of the aggregate and TrafficView the minimum of the generation times of the contained atoms, the parking guidance system compares aggregates by the *average* time of information generation of the atomic values contained in the aggregate. However, this again does not allow to decide reliably which of two aggregates is based on more precise data. Furthermore, it does not solve the duplicate problem, and hence does not allow merging aggregates for the same region. In this case our aggregation scheme could be applied as well, and would help to overcome these difficulties.

A substantial amount of work has been put into optimizing the beaconing in dissemination applications. This includes adaptive beaconing intervals, aiming to give cars with more or newer knowledge preferred access to the medium. It also encompasses the selection of the optimal subset of the knowledge base to be incorporated into a transmission of limited size. Work in this area can, for example, be found in [1, 2, 10, 17]. When speaking about applications for VANETs in a real environment questions about security arise. For instance, it has been studied how VANETs and aggregated messages in them can be made tamper-proof or resistant to cheaters [7, 8, 12, 15]. Since all these questions are largely orthogonal to the fundamental challenge of how aggregates should be computed, compared, and merged, their ideas can be combined with our approach.

In [3], Flajolet-Martin sketches are used for robust in-network aggregation in sensor networks in the presence of packet loss or node failures. The coordinated collection of information towards a sink in the sensor network is considered, as opposed to continuously updated distribution of information to all nodes as in this paper. The problem of removing old information, which we solve here, does therefore not arise.

In [16], the use of Flajolet-Martin sketches for spatio-temporal database indexes is discussed. The intention is to support queries of the form: "How many objects were in region x over the time interval t?" While these techniques could potentially be used to speed up specific queries in a centralized traffic information system, the problem is rather different from what we consider here, and their approaches are not transferable to a distributed, dissemination-based system.

Several other data structures for probabilistic, duplicate insensitive counting have been proposed lately. Examples are LogLog sketches [4] and the very recent HyperLogLog sketches [5]. Here, we use Flajolet-Martin sketches because they allow for a modification that enables the removal of old data from the aggregates.

3. ALGORITHM

3.1 Flajolet-Martin sketches

A Flajolet-Martin sketch (also called "FM sketch" or in the following simply "sketch") is a data structure for probabilistic counting of distinct elements that has been introduced in [6]. It represents an approximation of a positive integer by a bit field $S = s_1, \ldots, s_w$ of length $w \geq 1$. The bit field is initialized to zero at all positions. To add an element x to the sketch, it is hashed by a hash function h with geometrically distributed positive integer output, where $P(h(x) = i) = 2^{-i}$. The entry $s_{h(x)}$ is then set to one. (With probability 2^{-w} we have $h(x) > w$; in this case, no operation is performed.) A hash function with the necessary properties can easily be derived from a common hash function with equidistributed bit string output by using the position of the first 1-bit in the output string as the hash value.

The central result of [6] is that an approximation $C(S)$ of the number of distinct elements added to the sketch can be obtained by locating the end of the initial, uninterrupted sequence of ones, given by

$$Z(S) := \min(\{i \in \mathbb{N}_0 \mid i < w \wedge s_{i+1} = 0\} \cup \{w\}) \tag{1}$$

by calculating

$$C(S) := \frac{2^{Z(S)}}{\rho}, \tag{2}$$

with $\rho \approx 0.775351$.

The variance of $Z(S)$ is quite significant, and thus the approximation is not very accurate. To overcome this, instead of only one sketch a set of sketches can be used to represent a single value, trading off accuracy against memory. The respective technique is called Probabilistic Counting with Stochastic Averaging (PCSA) in [6]. With PCSA, each added element is first mapped to one of the sketches by using an equidistributed hash function, and is then added there. If m sketches are used, denoted by $S_1, \ldots, S_m$, the estimate for the total number of distinct items added is then given by

$$C(S_1, \ldots, S_m) := m \cdot \frac{2^{\sum_{i=1}^{m} Z(S_i)/m}}{\rho}. \tag{3}$$

But, as [6] also states, this formula is rather inaccurate as long as the number of elements is below approximately $10 \cdot m$. We thus modify (3) in the following way:

$$C(S_1, \ldots, S_m) := m \cdot \frac{2^{\sum_{i=1}^{m} Z(S_i)/m} - 2^{-\kappa \cdot \sum_{i=1}^{m} Z(S_i)/m}}{\rho}, \tag{4}$$

with $\kappa \approx 1.75$. This alleviates the initial inaccuracies, while otherwise being asymptotically equivalent to (3).

PCSA yields a standard error of approximately $0.78/\sqrt{m}$. For many VANET applications, sufficiently good approximations are possible at reasonable sizes.

Sketches can be merged to obtain the total number of distinct elements added to any of them by a simple bit-wise OR. Important here is that, by their construction, repeatedly combining the same sketches or adding already present elements again does not change the results, no matter how often or in which order these operations occur.

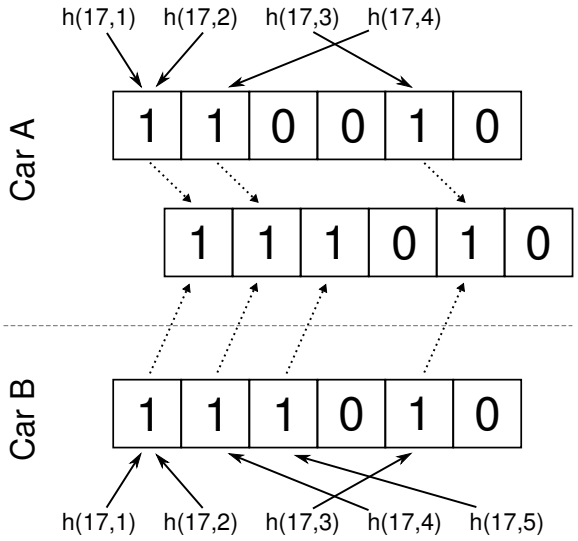

Figure 1: Generation and merging of FM sketch aggregates.

3.2 Creating and merging sketches

For the purpose of discussion, let us consider a specific application. Assume that we are interested in disseminating the number of free parking places. For now, we do not care about the measured values changing over time. As a first step, we use a sketch (or, with PCSA, a set of sketches) for each road segment. We assume that a car is able to observe the current number of free parking places while passing a road segment, e. g., by collecting data from sensors on the parking places, as proposed in [13]. After passing a road segment with ID r and observing n free parking places on it, a car may add the tuples $(r,1), \ldots, (r,n)$ to the sketch for r, by hashing them and setting the respective bits.

The locally stored sketches for the road segments are periodically broadcasted. Upon reception, received and local sketches are merged by calculating the bit-wise OR. Figure 1 exemplifies this procedure. Two cars, A and B, make independent observations on the same road segment (with ID 17). A observes 4 free parking places and thus hashes the tuples $(17,1), \ldots, (17,4)$ into its sketch for road segment 17. B observes 5 free parking places, and consequently adds $(17,1), \ldots, (17,5)$. If A and B meet later, and A receives a transmission containing B's sketch, A merges them by bit-wise OR and obtains a new sketch, replacing its previous one.

The hashed tuples (r,i) are identical for different observers, the observed value determines only how many of them are added. Thus the same number of free parking places on the same road segment will set the same bits, a lower number a subset thereof. Of course, in the current basic algorithm, bits that have once been set will never get unset again, and the sketch is therefore not able to follow decreasing values. We will soon see how to extend the data structure in order to overcome this limitation.

3.3 Hierarchical aggregation

Hierarchical aggregation is typically done on trees, often on symmetric and self-similar ones like quad-trees over the two-dimensional plane. But while it may for instance be expected that the traffic situation does not differ much among a set of similar and close-by road segments, or that the fraction of free parking places is relatively constant within a neighborhood, it might at the same time be vastly different not too far away: for example on the other side of a highway or a river. Therefore a good aggregation scheme

should respect the environment it is imposed on. We envision that such an aggregation hierarchy is pre-defined in the map data, following the underlying structure and grouping areas in a way that reflects their natural relations, like city districts or road hierarchies.

This is explicitly supported by our algorithm. Let L denote the set of locations for which observations can be made, like the entirety of road segments, or simply all points on the map. Many aggregates are possible—in principle, any arbitrary combination of locations could be aggregated. Possible aggregates are thus the (non-empty) elements of L's power set $\mathcal{P}(L)$. We may choose a subset $\mathcal{A}$ of these as the areas for which sketches are to be maintained:

$$\mathcal{A} \subseteq \mathcal{P}(L) \setminus \{\emptyset\} \tag{5}$$

The structure of $\mathcal{A}$ is not constrained by our method, though certain choices exhibit benefits.

Based on the outlined idea, hierarchical aggregation can be accomplished in the following way. We allot sketches for all elements in $\mathcal{A}$. Any observation made for some location $l \in L$ can immediately be incorporated into each aggregate for which the aggregated area A contains l, that is, $l \in A$. Consequently, in our example application of counting free parking places, these aggregates will contain the total counts in their respective areas.

It should be noted that not necessarily for each location a separate sketch needs to be maintained, i. e., there is no necessity that $\forall l \in L : \{l\} \in \mathcal{A}$. Especially if L is large (or continuous!), it might make sense to maintain sketches only for areas encompassing multiple locations.

Information on small-scale areas will then typically be kept in the closer vicinity and is broadcasted only there, while further away cars will preferably maintain and distribute larger-scale aggregates of the region. The duplicate insensitivity of the sketches allows for aggregates to be merged in just the same way as it has been introduced above for sketches of single locations. But in particular, any received sketch for some area A can immediately be incorporated into any *superordinate* aggregate A', where superordinate means that A is wholly covered by A', i. e., $A \subseteq A'$.

To allow for incorporating received information into as many aggregates as possible, $\mathcal{A}$ will thus indeed often be a hierarchical tree structure, where for all $A_1, A_2 \in \mathcal{A}$ it holds that

$$A_1 \cap A_2 \neq \emptyset \quad \Longrightarrow \quad A_1 \subseteq A_2 \ \vee \ A_2 \subseteq A_1. \tag{6}$$

Even this, however, implies neither any symmetry nor the same depth of all subtrees of any given node. It also does not exclude cases where some higher-level area is not completely covered by smaller subareas. Therefore, this concept is much more powerful than the commonly used aggregation trees.

3.4 Soft-state sketches

With the so far discussed algorithm, the sketches will always represent the maximum of all ever observed values for each road segment. This is of course not desirable. Therefore, a method is needed to remove old observations.

We accomplish that by modifying the original FM sketches. We use small counters of n bits length instead of single bits at each index position. These counters represent a time to live (TTL) in the range $0, \ldots, 2^n - 1$ for that bit. The operation of setting a bit to one after an observation is replaced by setting the corresponding counter to the maximum TTL, to $T := 2^n - 1$. Broadcasts containing the sketches are sent at regular intervals. Just before sending such a broadcast with information from the local knowledge base, all counters in the locally maintained sketches are decremented by one, if they are not yet zero.

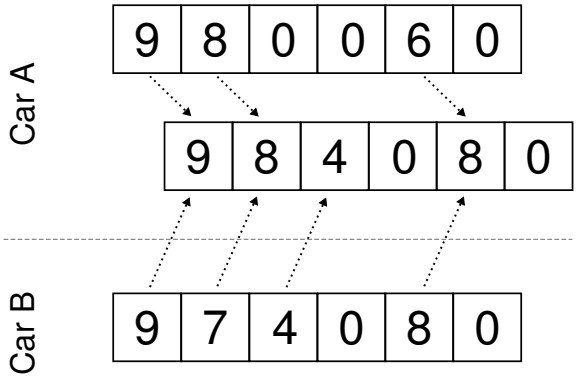

Figure 2: Merging of soft-state sketches.

When incorporating a received sketch into the local knowledge base, the bit-wise OR is substituted by a position-wise maximum operation. This yields a soft-state variant of FM sketches, in which previously inserted elements essentially die out after their TTL has expired, unless they are refreshed by a newer observation. The merging is visualized in Figure 2. Car A receives an aggregate from car B, and updates its own soft-state sketch accordingly.

For obtaining the current value from a soft-state sketch, the algorithm remains essentially unchanged; still, the smallest index position with value zero is identified and used. Note that this incurs some delay if a bit position is no longer set in newer observations. Coming back to our above example of observing parking places, assume that no further observations are made which set a particular bit position (e. g., because there is no longer a free parking place being hashed to it). If the position had previously been set in an aggregate, then the TTL value will decrease over time until it arrives at zero.

As an extension, it is possible to consider the smallest index position with a value below some threshold instead of the smallest position being zero, by using

$$Z_t(S) := \min(\{i \in \mathbb{N}_0 \mid i < w \wedge s_{i+1} \leq T - t\} \cup \{w\}) \tag{7}$$

in the role of $Z(S)$—which equals $Z_T(S)$—in (2) or (4). The threshold t can be chosen arbitrarily and individually whenever evaluating a sketch, in the range between 1 and T. This yields an approximation $C_t(S)$ of the total distinct item count observed in the last t broadcasting intervals. It therefore allows to dynamically choose a "cutoff horizon", thereby trading off between taking only more recent observations—that is, the most up-to-date data—into account and working on a larger data basis.

3.5 Longer counters for larger aggregates

In a typical application of our aggregation scheme, large aggregates will be distributed over longer distances, while smaller aggregates remain in the closer vicinity. But distribution over long distances implies that the aggregates are longer underway, and will typically traverse more hops. In short: the information will be older when it arrives at the place where it is used. Consequently, while locally relatively quick ageing of the information will be tolerable (or even desirable), for larger, widely distributed aggregates it will likely be advisable to use a longer lifetime for the soft-state information. To accomplish this, it is possible to extend our algorithm in a way that uses soft-state sketches with longer TTL counters for larger aggregates. Obviously, this increases the size of these aggregates, but since the available TTL range grows exponentially with the counter size it scales very well.

Using different counter lengths in different aggregates slightly increases the complexity of the merge operation. However, the necessary modifications become relatively straightforward if the *age* of an entry is considered instead of its TTL. The age is the number of decrements that have occurred at that position, i. e., its difference from the maximum TTL. Instead of setting the position in the merged aggregate to the maximum TTL of the two merged soft-state sketches, it is set to the TTL corresponding to the *minimum age*, where the received TTL is larger than zero. The resulting operation is equivalent in the case of identical counter sizes in the two sketches, but differs if the counter sizes are different.

As an example, consider a locally stored aggregate with a counter size of eight bits. Let us concentrate on one single position, and assume that it currently has a value of 8. This corresponds to an age of $255-8=247$, since the maximum TTL is $2^8-1=255$ here. Now a sketch for a sub-area contained in our aggregate is received, which uses a counter size of only four bits. The value at our bit position is 10 in this sketch. Because the maximum TTL is 15, this corresponds to an age of 5. Since the minimum age of local and received aggregate is 5, we set the local aggregate to this age, and consequently to a TTL value of $255-5=250$.

More formally speaking, if T_{local} and s_{local} are the maximum TTL and the current value of an entry of the locally stored sketch, and T_{recv} and $s_{\text{recv}} > 0$ are the corresponding counterparts in a received sketch that is to be merged in, then the new value of the respective position is given by

$$\begin{aligned} & T_{\text{local}} - \min\{T_{\text{local}} - s_{\text{local}},\ T_{\text{recv}} - s_{\text{recv}}\} \\ = \ & \max\{s_{\text{local}},\ s_{\text{recv}} + T_{\text{local}} - T_{\text{recv}}\}. \end{aligned} \tag{8}$$

This scheme is applicable for arbitrary combinations of counter sizes.

3.6 Example applications and practical issues

The simple parking place counting application as discussed above has a major drawback. In case of an aggregate showing a small total number of free parking places, it is not clear whether this is caused by a small number of parking places being free, or by a small number of free parking places having been *observed* in the time interval covered by the TTL, due to a generally low number of observations. Note that this is not a problem of our approach, but a general one. It is, fortunately, relatively easy to overcome. In the proposed application a car may distribute *two* values with separate sketches instead of just one: the number of observed free parking places, and in addition the number of observed total parking places. Both values in combination describe which *fraction* of the observed parking places is free. It generally seems that distributing such relative values is more robust. Recall that due to the soft-state approach it is not necessary to report occupied parking places since a no longer free parking place will die out if it is not refreshed.

This also accommodates trading off the considered timespan as introduced above: the application can easily infer the comprehensiveness and coverage of the underlying data basis for increasing time horizons, to optimize the tradeoff.

Sketches can be used to approximate sums of positive integers, but can be generalized to general integers and fixed or floating point numbers [3]. Our scheme is thus applicable whenever the aggregated value can be expressed through sums. Examples are counts, sums, or averages, but also variance and standard deviation (through the average and the average of the squares) or even products (by adding logarithms) [3]. The accuracies of the approximations, of course, vary, and an appropriate tradeoff for the specific application has to be found.

Further application examples could be the dissemination of the current traffic density (e. g., by distributing the number of observed vehicles and the total length of the roads for which there are observations), or the current average speed on a road. Both are useful to support navigation and route planning.

4. EVALUATION

4.1 Methodology

To evaluate our scheme, we implemented it in a simulation environment encompassing the network simulator ns-2 [11] and the microscopic traffic simulator VISSIM [14]. This combination [9] allows for a detailed simulation of both, vehicle movements and network traffic.

VISSIM was used to generate the vehicular movements. We used a detailed model of a real city's extended downtown area with more than 500 km of roads and up to 10 000 vehicles. It includes, for example, multi-lane traffic, traffic lights, and different types of vehicles, and simulates differing driver psychologies. The vehicular traffic in the model is based on extensive measurements undertaken by the city administration for the purpose of traffic planning.

We simulate a VANET equipment penetration ratio of 20 %. This corresponds to an average equipment density of one vehicle per communication range. In ns-2, IEEE 802.11 is employed as the MAC protocol, with the two-ray ground propagation model with a communication range of 250 meters and a carrier sense range of 550 meters. The network simulator is enhanced with an obstacle modeling that does not allow radio signals to propagate through the walls of buildings.

The road density, movement speed and pattern, etc. are, like in a real city, very heterogeneous in the simulation model. This makes the model realistic. But evaluating a protocol in such a complex environment in absolute terms is difficult. For example, if a high-speed road with lots of traffic connects two points that are 5 km apart, then one can expect well-working dissemination over that distance. If two other points with the same distance are separated by a municipal park, then the performance is bound to be worse. Often, it can make a large difference whether a single vehicle carrying a piece of information gets through to a certain point and distributes its knowledge there.

In order to overcome these difficulties at least partially, we opted to use an optimal reference protocol in an otherwise exactly identical simulation setting for comparison. In this protocol, information is spread by periodic beacons, in just the same way as in our and many other approaches. But the optimal protocol does not care about practical bandwidth limitations. In a simulator the packet size and the amount of information actually exchanged between the nodes are independent. The reference protocol exploits this fact and does *not* aggregate information at all. Instead, a received "aggregate" contains the sending node's most up-to-date measurement values of all the locations. Each of these values also carries an individual timestamp, making optimal merging of information trivial. In order to avoid keeping too old measurements in the network, we use a timeout equivalent to the initial TTL of the soft-state sketches' entries.

Obviously, implementing this optimal reference protocol is easily possible in a simulator, but not in practice. However, it is well suited as a benchmark: with a practical protocol based on beacons and a knowledge base the cars can never have better information.

We evaluate both our scheme and the optimal reference protocol with an idealized application. We subdivide the city area into 256 small areas, which we use as single locations. We simulate a simple stochastic process for each of these areas, the current value of

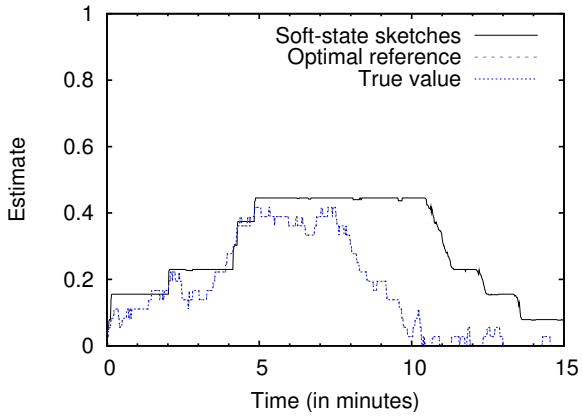

Figure 3: Accuracy of local measurement representation.

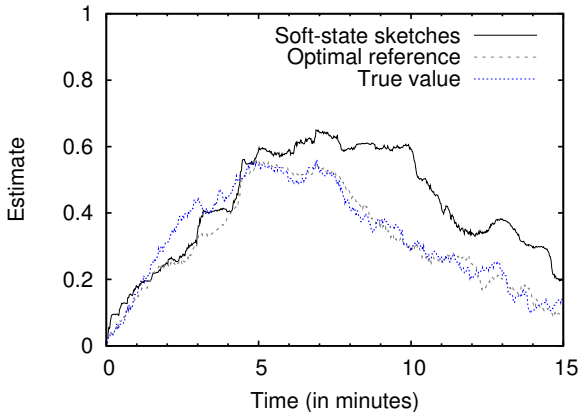

Figure 4: Accuracy of local medium-sized aggregate.

which can be "measured" by every car entering the respective area. This could be interpreted as "counting" the number of free parking places in this area.

Our simulations cover a timespan of 15 minutes. As aggregation generally makes sense only if the dynamics at least of close-by areas is not completely unrelated (which can of course be expected in the real world), the stochastic processes of the single locations are also not totally independent. All of them typically start relatively low, and increase substantially over the first half of the simulation, before they subsequently tend to decrease again. In relation to the relatively short total time, the changes are very rapid, and thus are very challenging for a dissemination protocol that needs to keep track of them. Since, as discussed above, disseminating relative values is much more appropriate than absolute sums, we also have one reference value per area—e. g., the "total number of parking places"—, which is likewise "observed" and distributed.

For hierarchical aggregation, groups of four neighboring areas are combined into a total of 64 medium-sized areas, and these are in the same way aggregated further to form 16 large-scale areas. For deciding which information is to be transmitted in a beacon, we constrain ourselves to a simple selection strategy—mainly to limit the overall complexity of the system and to allow for reasonable result interpretation. In our simulations, each node transmits 27 aggregates in each beacon: on all three hierarchy levels, the area the car is currently in and the eight neighboring areas are sent.

4.2 Local accuracy

The first parameter we evaluate is how well the aggregates reproduce the current value of a measured parameter *locally*, i. e., at the location where the measurement is performed. This is of interest because the data representation with soft-state sketches is probabilistic, and therefore does not necessarily reproduce the current value exactly.

Figure 3 shows, for one typical location, the average value from the knowledge bases of all vehicles that are currently within the corresponding region. As discussed above, we use a relative value, given by the ratio of the number currently free "parking places", which is changing over time, and their (fixed) total number for this location. For comparison, we also plot the current true value, and we perform the same evaluation with the optimal dissemination protocol. In the specific case of a single, locally continuously measurable value the latter two are of course always identical.

We use PCSA sets with 16 soft-state sketches and 5-bit counters in the evaluations shown here. The maximum TTL of the soft-state sketch entries is set to 31, the beaconing frequency is five seconds. The figure shows that the sketch-based dissemination is able to model the correct value quite well. When the measured value starts decreasing again, the time needed for the soft-state decaying of the no longer set bits becomes visible: the sketch represents the maximum value observed in the recent past. Thus, there is a time lag before it follows a decreasing value, about $31 \cdot 5\,\mathrm{s} \approx 2.5\,\mathrm{min}$ with the chosen parameters. Recall in this context that an application can dynamically select the cutoff horizon in (7), and may therefore tune this parameter locally and individually at any time.

Taking the time lag into account, the soft-state sketches indeed reflect the true situation very well here. This encouraging first result leads us to the next question: how well can aggregates for larger regions be formed and maintained?

4.3 Forming aggregates

With the proposed scheme, aggregates for larger regions can and will be formed wherever information on the respective region flows together. But nevertheless it is reasonable to expect that this will most regularly happen within the respective region. Therefore, we now look at how well sketch-based aggregates represent the situation in a larger area, while they are stored, passed around, and merged by cars within this area. Note that now the cars can no longer observe the entirety of the underlying information themselves. They can only measure the current value of their own location, and thus depend on received and merged information from other nodes in order to complete their picture.

In Figures 4 and 5 we show such evaluations for a typical medium-sized and a large inner-city aggregate (consisting of 4 and 16 locations, respectively). It is visible that perfect knowledge like in Figure 3 is no longer possible: the optimal reference protocol does not always have perfect information. Especially for the high-level aggregate in Figure 5 the inevitable delay until up-to-date information on the entirety of the area has arrived can clearly be seen from the offset between the true value and the optimal protocol. The sketch-based aggregation is close to the optimal knowledge, again with the previously discussed soft-state time lag in case of a decreasing value. These results demonstrate that probabilistic aggregation is indeed able to collect the available information, yielding aggregates that represent the available knowledge.

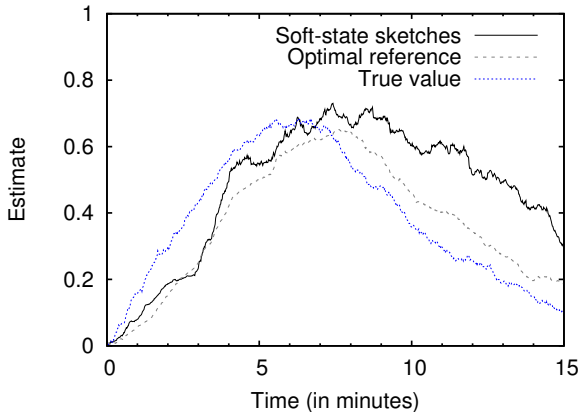

Figure 5: Accuracy of local large aggregate.

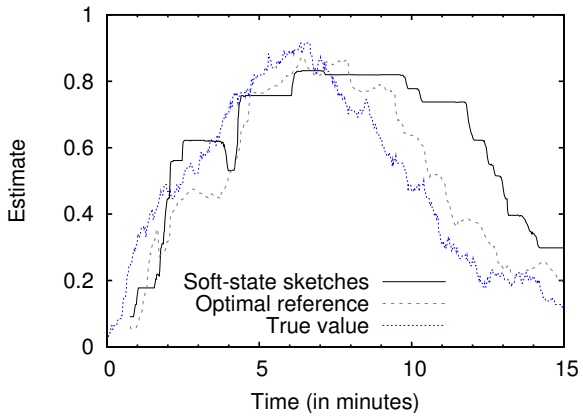

Figure 6: Accuracy of distant medium-sized aggregate.

4.4 Distributing aggregates

Now, we look at further away regions and evaluate the representation of the situation in the knowledge bases of participating cars. Figures 6 and 7 represent two instances of medium-sized regions, as they are seen at a distance of about 3 km beeline. In addition to the time lag for the information transport it is not astonishing that even the optimal reference protocol does not always have complete information. If up-to-date data only on parts of the total aggregation area is present and this data is not typical for the whole aggregate, effects like the overestimation around simulation minute six in Figure 7 are the logical consequence. Nevertheless, the estimates once again reflect the true situation in the modeled regions well. Finally, this is also confirmed by a look at a large aggregate's representation of the outskirt area in our model, as it is depicted in Figure 8.

In summary, sketch-based probabilistic aggregation can be used to create aggregates that come close to what can theoretically be achieved with the considered kind of system. This optimum was here represented by a non-realizable optimal reference protocol. We consider these results very encouraging indications that the proposed algorithm is a suitable way to overcome the often observed general difficulties of distributed, uncoordinated data aggregation in dissemination schemes.

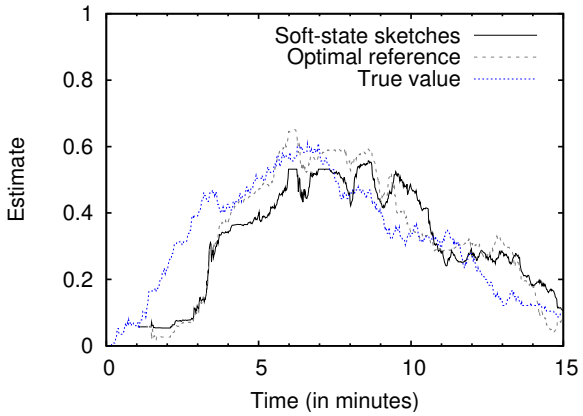

Figure 7: Accuracy of distant medium-sized aggregate.

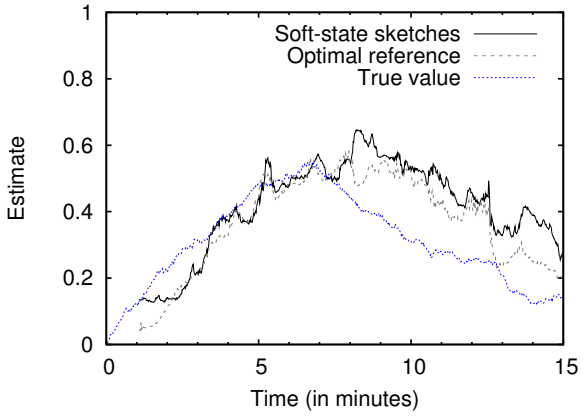

Figure 8: Accuracy of distant large aggregate.

5. CONCLUSION

In this paper, we introduced a data aggregation mechanism for dissemination-based VANET applications. It is based on a probabilistic data representation, Flajolet-Martin sketches, which we extended to a soft-state data structure. This yields duplicate insensitive aggregates, thereby overcoming major unsolved problems in existing aggregation schemes for VANETs. We discussed several application scenarios of our scheme and possible extensions, showing the broad applicability of the approach.

In a simulative evaluation study using ns-2 and a realistically modeled inner-city VANET we assessed the performance of the aggregation scheme. We compared the values to what could ideally be achieved in the same setting. The results of this study confirm our expectation that sketch-based schemes are well-suited for the considered purpose, and may well form a central building block for a large variety of future VANET applications.

6. REFERENCES

[1] C. Adler, R. Eigner, C. Schroth, and M. Strassberger. Context-adaptive information dissemination in VANETs – maximizing the global benefit. In *CSN '06: Proceedings of the 5th IASTED International Conference on Communication Systems and Networks*, pages 7–12, Aug. 2006.

[2] M. Caliskan, D. Graupner, and M. Mauve. Decentralized discovery of free parking places. In *VANET '06: Proceedings of the 3rd ACM International Workshop on Vehicular Ad Hoc Networks*, pages 30–39, Sept. 2006.

[3] J. Considine, F. Li, G. Kollios, and J. Byers. Approximate aggregation techniques for sensor databases. In *ICDE '04: Proceedings of the 20th International Conference on Data Engineering*, pages 449–460, Mar. 2004.

[4] M. Durand and P. Flajolet. Loglog counting of large cardinalities. In *ESA '03: Proceedings of the 11th Annual European Symposium on Algorithms*, pages 605–617, Sept. 2003.

[5] P. Flajolet, É. Fusy, O. Gandouet, and F. Meunier. Hyperloglog: The analysis of a near-optimal cardinality estimation algorithm. In *AofA '07: Proceedings of the 2007 International Conference on Analysis of Algorithms*, June 2007.

[6] P. Flajolet and G. N. Martin. Probabilistic counting algorithms for data base applications. *Journal of Computer and System Sciences*, 31(2):182–209, Oct. 1985.

[7] P. Golle, D. Greene, and J. Staddon. Detecting and correcting malicious data in VANETs. In *VANET '04: Proceedings of the 1st ACM International Workshop on Vehicular Ad Hoc Networks*, pages 29–37, Oct. 2004.

[8] M. Jakobsson and S. Wetzel. Efficient attribute authentication with applications to ad hoc networks. In *VANET '04: Proceedings of the 1st ACM International Workshop on Vehicular Ad Hoc Networks*, pages 38–46, Oct. 2004.

[9] C. Lochert, M. Caliskan, B. Scheuermann, A. Barthels, A. Cervantes, and M. Mauve. Multiple Simulator Interlinking Environment for Inter Vehicle Communication. In *VANET '05: Proceedings of the 2nd ACM International Workshop on Vehicular Ad Hoc Networks*, pages 87–88, Sept. 2005.

[10] 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*, 8(3):6–19, 2004.

[11] The Network Simulator ns-2. http://www.isi.edu/nsnam/ns/.

[12] F. Picconi, N. Ravi, M. Gruteser, and L. Iftode. Probabilistic Validation of Aggregated Data in Vehicular Ad-Hoc Networks. In *VANET '06: Proceedings of the 3rd ACM International Workshop on Vehicular Ad Hoc Networks*, pages 76–85, Sept. 2006.

[13] M. Piorkowski, M. Grossglauser, and A. Papaioannou. Mobile user navigation supported by WSAN: Full-fledge demo of the SmartPark system. In *MobiHoc '06: Proceedings of the 7th ACM International Symposium on Mobile Ad Hoc Networking and Computing*, May 2006. Technical Demonstration.

[14] PTV AG. VISSIM. http://www.english.ptv.de/cgi-bin/traffic/traf_vissim.pl.

[15] M. Raya, A. Aziz, and J.-P. Hubaux. Efficient Secure Aggregation in VANETs. In *VANET '06: Proceedings of the 3rd ACM International Workshop on Vehicular Ad Hoc Networks*, pages 67–75, Sept. 2006.

[16] Y. Tao, G. Kollios, J. Considine, F. Li, and D. Papadias. Spatio-temporal aggregation using sketches. In *ICDE '04: Proceedings of the 20th International Conference on Data Engineering*, pages 214–226, Mar. 2004.

[17] L. Wischhof, A. Ebner, and H. Rohling. Information dissemination in self-organizing intervehicle networks. *IEEE Transactions on Intelligent Transportation Systems*, 6(1):90–101, Mar. 2005.

[18] L. Wischhof, A. Ebner, H. Rohling, M. Lott, and R. Halfmann. SOTIS – a self-organizing traffic information system. In *VTC '03-Spring: Proceedings of the 57th IEEE Vehicular Technology Conference*, pages 2442–2446, Apr. 2003.