Reducing Redundant Data Transmissions in Wireless Ad Hoc Networks: Comparing Aggregation and Filtering

Ouksel, Aris and Lundquist, Doug

Department of Information and Decision Sciences,
University of Illinois at Chicago,
Chicago, Illinois, USA

E-mail: aris@uic.edu, dlundq1@uic.edu
Phone: 312-996-0771
Fax: 312-413-0385

Abstract—Efficient bandwidth usage is vital for real-time ad hoc networking applications like vehicular safety. Yet, such applications can produce large amounts of identical data. Pruning redundant data transmissions can enable delivering richer data to more users at shorter intervals. Reducing redundancy has been studied extensively for stable network topologies but the solutions are not directly extensible to dynamic topologies where information about network state obsolesces quickly. We compare two novel combinations of the adaptive controlled flooding routing protocol, SBSF, with implementations of response aggregation and query filtering for mobile environments. We test these combinations in simulated vehicular networks. We show that, even in cases where response aggregation only slightly improves network performance, query filtering can improve delivery by up to 30% and response time by 75%.

Keywords—*Mobile computing, broadcasting, vehicular networking, simulation*

1 Introduction

Ad hoc networks have long been envisioned for areas lacking communication infrastructure.

However, recent growth in mobile computing has driven wider infrastructure availability. In areas with 3G or 4G service, many proposed peer-to-peer applications for video streaming and travel information have been rendered obsolete. Yet, ad hoc networking can still play the primary role in certain applications, such as real-time video streaming for vehicular safety [34] and, more generally, situational awareness (SA) applications for mobile environments.

In essence, SA maintains a complete view of a set of environmental variables, including expected future changes. Accurate SA enables preventing problems rather than addressing them after they occur but can be demanding in terms of network transmission capacity. The faster conditions change on the ground, the faster information must be provided to network users. Likewise, more complex situations require monitoring larger sets of variables.

Many SA domains are potentially both complex and highly dynamic. An early, important paper on vehicular ad hoc networks (VANETs) considered sending queries about local travel conditions to roadside infrastructure [11]. Another example is assisting military operations; at the tactical level, information about friendly and hostile forces must be updated continuously. Other SA domains include: disaster recovery [19], where resources must be allocated sequentially to save lives and minimize property damage; dynamic pricing of toll roads and parking spaces; and even advertising strategies for yield management [20]; and vehicular safety [7].

This paper considers the unique problems of vehicular safety applications, which must continuously gather and disseminate detailed information about all vehicles in an area. Consider collision warnings for vehicle streams approaching a curve from opposite directions. Naïve, straight-line extrapolation of vehicle trajectories will generate many spurious collision alerts. Instead, safety alerts might watch for vehicles drifting out of their lanes, indicating driver inattention, intoxication, or incapacity. To reduce the impact of observation errors, alerts would include perspectives of multiple observers. We contend that, in dense environments, vehicular safety updates will require shortened transmission ranges and multi-hop communication and also generate data that is highly prone to redundancy.

Short transmission range: Shorter range lets nodes transmit more often and, given sufficient node density, improves network throughput. The two basic strategies for dealing with high node density are shortening transmission range [36] or slowing the transmission rate [12], [32].

Vehicular safety requires a minimum acceptable notification rate; the standard transmission ranges provided by 802.11p – which can at times encompass a thousand or more vehicles [1] – may be too long. Indeed, a majority of beacons will be lost at a fraction of that density [4]. Broadcast storm can also arise from transmissions by roadside infrastructure, as considered in [35].

Multi-hop Communication: With short transmission range, it follows that SA updates must be forwarded over multiple hops to inform distant network users. For example, suppose an erratic driver is detected. If warnings are ultimately forwarded to vehicles a few kilometers away, they will be better prepared to avoid collisions.

Redundancy: Many readings from different observers may be identical. Such readings can be combined without loss and pruning their redundancies mitigates broadcast storm. As a network policy, reduced redundancy could allow more frequent updates, a larger set of observed variables, or even reduced power consumption.

Many methods have been proposed for reducing redundant data transmission in low-mobility networks. In various ways, they avoid sending the same data over the same route more than once. High-mobility networks – such as vehicular ad hoc networks – often have ephemeral routing paths, making it impractical to obtain the knowledge required by such methods. Further, extant methods for high-mobility networks minimize bandwidth consumption to fulfill a given set of queries or notifications. While such methods indeed free up bandwidth for additional network traffic, they lack mechanisms to adapt the SA updates rate to network activity.

We propose a different approach: increasing data throughput to allow SA updates over longer distances and at shorter intervals. In [25], the Self-Balancing Supply/Demand (SBSD) controlled flooding protocol showed effectiveness in dynamic topologies. Here, we extend SBSB with response aggregation and query filtering (defined in the next section), both separately and together. We test these extensions in a simulated vehicular network.

The remainder of this paper is ordered as follows. In section 2, we define four basic methods for reducing redundancy. In section 3, we cover relevant research. In section 4, we briefly describe SBSB and detail our two redundancy reduction models. Finally, we present our simulation results in section 5.

2 Methods for Reducing Redundant Transmissions

We illustrate four basic methods for reducing redundant transmissions of data using a scenario of four cars (A, B, C, and D) driving in series (Fig. 1). To better illustrate these methods, we assume a sparse vehicle distribution such that each car is only in communication range of the ones immediately in front of or behind it; *e.g.*, C can communicate with B and D but not A. We assume other vehicles exist in both directions; D can communicate with them through a car E and A can through a car Z. A query for mobility data from a set of cars X is expressed as $q(X)$ and the

response containing mobility data from a set of cars X is expressed as $r(X)$. For example, in regard to a set $X = \{A, B, C\}$, the query would be $q(A, B, C)$ and the response would be $r(A, B, C)$.

Fig. 1. Vehicle configuration.

Data aggregation constructs aggregated packets from two or more packets with duplicate data.

Definition: Given a node n holding two responses r_1 and r_2 , with $(r_1 \cap r_2 \neq \emptyset)$, data aggregation will have n construct an aggregated packet containing $(r_1 \cup r_2)$.

Example: Car A receives two queries $q(B, C)$ and $q(C, D)$ via Z. After receiving $r(B, C)$ and $r(C, D)$, instead of forwarding them separately to Z, A sends $r(B, C, D)$.

Demand aggregation delivers packets to multiple recipients along a common routing path, rather than constructing each path separately.

Definition: Given a response r to be routed to two different destinations over two sets of nodes P_1 and P_2 . If $(P_1 \cap P_2) \neq \emptyset$, any nodes in $(P_1 \cap P_2)$ need only transmit r once for it to be delivered to both destinations.

Example: Cars A and C have both sent $q(D)$ to D. If D knew A was reachable via C and B, it could send $r(D)$ to A via C and knowing that both would receive it.

Query filtering reduces the scope of queries to exclude information that is already held or for which a query has already been forwarded.

Definition: Given a node n that forwarded a query q and later receives a query q' , $(q \cap q') \neq \emptyset$, query filtering will rewrite q' to request only $(q' - (q \cap q'))$.

Example: Car C holds $r(C)$ but not $r(D)$ and receives $q(C, D)$ from A. C only forwards $q(D)$ to D, as it can already fulfill $r(C)$ itself.

Subsumption is a query (or response) being wholly contained within a set of queries (or responses). Exact identification of subsumption relationships can be impractical since solutions

are NP-complete [29]. An alternative is probabilistic subsumption-checking, such as [24], which samples points from the query space (and which we describe further in 4.3).

Definition: Given a set of n queries $Q = \{q_1, q_2, \dots, q_n\}$, $n \geq 1$, then a query q is said to be subsumed by Q if $q \subseteq \{q_1 \cup q_2 \cup \dots \cup q_n\}$. Similarly, given a set n responses $R = \{r_1, r_2, \dots, r_n\}$, if $r \subseteq \{r_1 \cup r_2 \cup \dots \cup r_n\}$ then r is subsumed by R .

Example: Car A has just sent its own query $q(C, D)$ to B. A then receives $q(C)$ via Z. A does not forward $q(C)$, as A will receive $r(C)$ subsumed in $r(C, D)$.

3 Related Research

Stable network topology and accurate knowledge of query and data distributions reduce the need for redundant routing and caching. Stable topology allows reliable single-path routing. Known query distributions let data be stored near likely requesters. Known data locations eliminate the need for search within the network. The topological stability of wired networks enables efficient delivery methods and architectures. Google, for example, tracks query distributions to cache locally popular Web pages at its servers. This speeds response times and reduces forwarding queries to other clusters. Google also continually refreshes caches, as query popularity can decline quickly. Developing similar predictive and adaptive methods for wireless ad hoc networks must consider node mobility.

	Low Absolute	High Absolute
Low Relative	<i>Network configurations:</i> Static sensor networks; Pedestrians with mobile devices <i>Concerns:</i> Preserving battery life; Providing fast responses	<i>Network configurations:</i> VANETs (constrained by roads): congested traffic, one-way travel <i>Concerns:</i> Maintaining routing trees; Access to fixed wireless infrastructure
High Relative	<i>Network configurations:</i> Mixed-mobility networks: Mostly pedestrians, some vehicles <i>Concerns:</i> Preserving battery life; Routing paths often unknown	<i>Network configurations:</i> VANETs (constrained by roads): Two-way travel Mixed-mobility networks: Mostly vehicles, some pedestrians <i>Concerns:</i> Routing paths very often unknown

Table 1. Classification by mobility.

Wireless ad hoc networks can be categorized by *absolute* mobility, node speeds, and *relative* mobility, the speeds at which nodes move relative to each other. Networks with low absolute and relative mobility have time to gather network knowledge; those with high absolute and relative mobility do not. We classify typical examples of mobility combinations in Table 1, below, before elaborating on their redundancy reduction methods in the next two sections.

3.1 Redundancy Reduction in Low Relative Mobility Networks

Many readers may be familiar with data aggregation in low-mobility sensor networks, such as [27]. We discuss some representative methods to illustrate which can (and cannot) be applied to constrained mobility VANET scenarios. Typically, low-mobility sensor networks have stable routing paths and nodes have fixed roles as either data sources or sinks. Methods from wired computing [6] can substantially reduce bandwidth and power usage. The recent PhotoNet [30] removes redundant frames from streams of visual data to deliver diverse content. The model includes additional features to reduce redundant transmissions – for example, upon encountering each other, nodes exchange lists of the photos they have recently seen. However, the paper only gives results for low-density, intermittently connected networks. It is unclear if the model as presented would perform well in dense, dynamic networks using broadcasting.

For large-scale, interconnected sensor networks, query filtering and subsumption can reduce network traffic by an order of magnitude [15], [16], [18]. However, filtering policies must be carefully chosen for specific networks and applications. Filtering at query sources reduces network traffic the most but sends more disjoint queries, with fewer opportunities to aggregate data or demand. Filtering at intermediate nodes can help cache popular information near future queries and lessens the risk that any recent updates will be missed.

3.2 Redundancy Reduction in High Relative Mobility Networks

In networks exhibiting high relative mobility, the topology, query distribution, and locations of data may be largely unknowable. It is difficult to optimize what data to cache, where to cache it, and when to discard it. This increases the costs of search and dissemination within the network.

For routing, networks must also rely more on flooding instead of learning and re-using routing paths. Researchers have nevertheless devised predictive caching models for such dynamic environments.

The seminal paper [14] proposed three methods (based on access frequency and network topology) for predictive data caching against anticipated demand. A general-purpose method for proactively pushing information to meet its anticipated demand is presented in [13]. Anticipated demand is based on historical trends, so their approach depends on stable query distributions and, to some extent, on a number of queries being very popular. The SBSR routing protocol caches data inherently by responding to queries, as matching responses propagate within each query's flooding area.

Demand aggregation is also useful in VANET applications; an early but well-known example using tree-based routing given in [5]. While sensor networks typically deliver information from many sensors to few sinks, VANETs often deliver identical information to many vehicles (*e.g.*, upcoming travel conditions to a set of approaching vehicles). Further, in sensor networks, the content, timing, and volume of messages can be regulated by network policy. Demand aggregation in VANETs is complicated by privacy concerns and intermittent partitioning, particularly in applications designed for use by the general public.

Some researchers have addressed data redundancy in VANETs by combining similar information more compactly, *e.g.*, by averaging or sampling. A framework for sharing aggregate historical information among vehicles is given in [9] but does not include performance results. Probabilistic aggregation of binary state information for sets of identical items was examined in [21] but is not extensible to non-identical items. Even non-redundant data may be transmitted more efficiently by grouping related items, such as all vehicular safety updates generated during a short interval; this approach was observed in [33] to beneficially impact network throughput.

Discarding highly similar data has also been applied to VANETs. In [10], a fuzzy logic model evaluates the similarity of data and avoids transmitting similar content. However, it is intended for applications not requiring exactness. Another approach, from [23] proposes a multi-level model that separates received data into more fundamental units to facilitate eliminating redundant

transmissions. While their model enables more efficient bandwidth usage, it also incurs substantial overhead in processing and transmissions; its suitability for dense networks with high relative mobility is unclear.

On highways, vehicles' relative positions tend to be stable in regard to same-direction travel, letting VANETs apply data aggregation and query filtering models from sensor networks. For example, in [37], nodes delay forwarding packets in order to limit redundant transmissions. The expectation is that waiting will allow nodes to receive identical data that can be combined. Of course, vehicular safety information is generally much more time-sensitive than broader traffic conditions. A two-tiered information delivery platform (for short- and long-range SA) is proposed in [15]. This model has each node broadcast its information about nearby vehicles using a combination of data compression and aggregation.

The response aggregation and query filtering models we combine with SBSA differ from the preceding models as follows (and which will be detailed in section 4). First, they require no coordination among nodes; this makes them suitable for dynamic environments where coordination is problematic. Second, they do not inherently require any delays to accumulate redundant data; this makes them suitable in time-sensitive applications like vehicular safety. Finally, in the response aggregation model, each node's decisions are determined according to its local collision rate, letting it adapt to changing network conditions.

4 Model Implementation

In this section, we present our two redundancy reduction models. In section 4.1, we describe SBSA, which is necessary to our presentations, in sections 4.2 and 4.3, of our response aggregation model and our probabilistic subsumption-based query filtering model. Finally, section 4.4 gives an estimation model for their consequent performance gains, in terms of flooding depth and query fulfillment.

4.1 Self-Balancing Supply/Demand

The basic life cycle of a query in SBSB, from posting to expiration, is as follows. For a node n posting a query q , q is flooded via broadcast around n (we describe SBSB's adaptive flooding depth in the following paragraphs). When a replica of q is received by a node n' that holds a response r matching q , n' creates a matched query q' by appending r to q . This matched query is then flooded back to n .

Flooding depth in SBSB is variable and adapts to the volume of network traffic. SBSB uses the congestion metric *utility* to rank packets and thereby regulate flooding depth. It is applied in the same fashion to queries and responses (which are stored as data fields appended to query packets). A query replica's utility u at a node n is given by Eq. 1 below, where a is its age (the elapsed time since the original query was posted), h is the number of hops it replicated to reach n , and f is its frequency (the number of nodes that have posted the same query, as known by n).

$$u = -\frac{ah^{1/2}}{f^{1/3}} \quad (1)$$

Utility represents the negative ratio of the congestion a query has inflicted to the congestion it is allowed to inflict. Inflicted congestion is estimated from the time and distance a query has flooded from its source, increasing with both. Allowed congestion is not calculated directly but, as a policy, scales linearly with frequency; more popular queries flood farther and receive more broadcasts at each node. This scaling is a consequence of the above utility function. As more queries flood an area, each one's allowed congestion is smaller. This system behavior arises from nodes only broadcasting their high-utility replicas, as follows.

Each node independently determines u_{min} , the minimum utility at which a replica may be forwarded. This u_{min} is calculated after each broadcast by forming the set of highest-utility packets a node expects to be able to transmit at a binary exponential rate for their remaining times-to-live. As long as a replica's utility remains above u_{min} , it may be repeatedly broadcast, giving robustness against collisions and temporary partitions. Since SBSB uses flooding, nodes near each other tend to experience similar network traffic and thus have similar u_{min} values. This policy induces a predictable flooding depth that adapts to changing network traffic levels.

The standard model of SBSB uses a context-aware cross-layer (CACL) medium access control compared against the 802.11 MAC in [22]. From CACL, SBSB receives information about node density to determine u_{min} and schedule repeated broadcasts of packets, which in turn drives the packets CACL receives from SBSB. CACL remedies limitations of the 802.11 MAC for broadcasting in dense environments, like hidden terminal collisions and nodes claiming the channel for excessively long periods. The basic features of SBSB using CACL are given below.

Density estimation: Each node learns its one-hop neighborhood N_1 by received packets. All sources of received transmissions in the previous 500ms are included in N_1 . For vehicular applications, the two-hop neighborhood node population N_2 is estimated as $2N_1$. Density estimates are used to schedule broadcasts by imposing post-broadcast delays and delays between repeat broadcasts of a packet.

Channel sensing: To avoid single-hop collisions, nodes do not broadcast while the channel is in use. Instead, a node scheduled to broadcast but finding the channel busy checks at random intervals (averaging 0.1ms) for the channel to be free.

Post-broadcast delay: After a node n broadcasts, it incurs a variable delay d before it can broadcast again. Given N_2 and packet transmission time t , d is randomly taken from the uniform distribution $[(N_2 - N_1)t, 3(N_2 - N_1)t]$.

Collision rate: The above post-broadcast delay gives an expected collision rate of $\frac{1}{e} \approx 0.367$. This is a well-known result for a group of k nodes each broadcasting once during a set of k frames and its proof is therefore omitted.

Repeat broadcasts: Repeat broadcasts occur at a binary exponential rate with base $2N_2t$. To correct for changing density, a packet's base is updated after it is broadcast.

Packet selection: For its next scheduled broadcast, a node n selects a packet from the set having utility $> u_{min}$. The packet with the fewest prior broadcasts by n is chosen. If multiple packets thus qualify, the highest-utility one is chosen.

4.2 Response Aggregation Model

Our aggregation model permits pairwise aggregation of packets containing common data without loss. For any two sets of response data A and B , we define *overlap* as occurring when $(A \cap B) \neq \emptyset$. Measuring A , B , and $(A \cup B)$ in terms of data size, we define the magnitude ω of any overlap as $[A + B - (A \cup B)] / (A \cup B) = (A \cap B) / (A \cup B)$, which scales to $0 < \omega \leq 1$. If $\omega = 0$, $(A + B) = (A \cup B)$; A and B have no data in common. If $\omega = 1$, A and B contain identical response data; their intersection equals their union.

When a node's upcoming broadcast contains response data, it may combine the *primary packet* (already selected for broadcast) with a *secondary packet* (which contains some of the same data as the primary packet). Ensuring aggregation is beneficial requires estimating the net benefit of aggregating any two responses. We use a benefit ratio β to compare the additional response data transmitted per unit time against any expected increase in the collision rate. When a node determines its best benefit ratio $\beta_{max} > 1$, aggregation is expected to be beneficial. Else, the primary packet is broadcast without aggregation. From its packets having utility $> u_{min}$, n selects the secondary packet yielding the highest β as follows.

$$\beta = \frac{\Delta\tau}{\Delta c} \text{ where:}$$

$\Delta\tau$ (the change in throughput): Given two responses of sizes A and B having overlap ω , with $0 < \omega < 1$, aggregation raises throughput by a factor of $1 + \omega$. In effect, during the time required to transmit data of size $(A \cup B)$, in terms of query fulfillment $(1 + \omega)(A \cup B)$ is being transmitted. Thus, $\Delta\tau = 1 + \omega$.

Δc (the change in collisions): We model the cost of collisions as a linear function of the size of aggregated packets. This captures the effect of data loss being proportional to packet size. As CACL is designed to incur a $\frac{1}{e}$ collision rate, we assume future collision rates will be at least $\frac{1}{e}$. For a primary packet A and secondary packet B , we define $\Delta c = \frac{1}{e} \frac{A \cup B}{A}$. Note that Δc increases when the overlap ω increases and when A is smaller than B (even if A is subsumed within B). The latter avoids transmitting large aggregated packets at the short intervals at which CACL will transmit small primary packets.

Our aggregation model anticipates that nodes may experience collision rates that differ from the norm or change over time. Variable transmission ranges (used by this paper’s simulations for improved realism) reduce CACL’s expected $\frac{1}{e}$ collision rate. Since N_2 as estimated by a node n includes nodes beyond the average transmission range, on average it overestimates the number of nodes expected to cause collisions with a given transmission by n .

To address uncertainty in collision rates and their measurement, each node tracks its local collision rate using an exponential smoothing model. Our goal is simply to estimate the future collision rate based on recent evidence, in order to adapt to changes in nearby transmission activity. After every collision or successful receipt, each node updates its estimated receipt rate s from its previous estimate s_0 using the formula $s = (1 - \alpha)s_0 + \alpha x$, where x is 1 for a successful receipt and 0 for a lost packet. The estimated collision rate c is then $(1 - s)$. We set $\alpha = 0.02$, which means that half of c ’s value is determined by the node’s previous 34 packets (*i.e.*, $(1 - \alpha)^{34} \approx 0.5$). In practice, nodes would adapt to a changed collision rate within a few seconds.

4.3 Subsumption-Checking Model

We now present our subsumption-checking model, using the scenario in Fig. 2, below.

Fig. 2. Mapping queries and responses.

Scenario: An SA application operates in area S . Four network users (Albert, Bob, Chuck, and Don) respectively post queries for areas A , B , C , and D within S . These queries are mapped to S as shown in Fig. 2.

Query example: Queries $q(A)$ and $q(B)$ seek current vehicle positions and speeds in rectangular areas A and B , respectively.

Response example: Responses $r(A)$ and $r(B)$ are the matching mobility data from the two areas.

Overlap example: The *overlap* of $r(A)$ and $r(B)$ is any data in intersection $A \cap B$.

Consider Don's query $q(D)$, which is subsumed by $q(A) \cup q(B)$ but by neither $q(A)$ nor $q(B)$ alone.

This gives four possibilities. First, if Don already has $r(A)$ and $r(B)$, $q(D)$ can be fulfilled locally without sending it to other nodes. Second, if Don receives $r(A)$ and $r(B)$ while waiting for $r(D)$, he can construct $r(D)$ from them. Third, if a node E near Don has received $q(D)$, $r(A)$, and $r(B)$, E can construct $r(D)$ and forward it to Don. Finally, if E has already forwarded $q(A)$ and $q(B)$ before receiving $q(D)$, E might not forward $q(D)$ if it expects to receive both $r(A)$ and $r(B)$.

These possibilities can be generalized as follows. Assume every node n has a cache of Q_n queries and R_n responses which we term n 's queue. Consider a query q with a hypothetical matching response r . Node n could check for subsumption when:

- **Posting q .** Here, n checks if $r \subseteq R_n$. If so, q would not be transmitted to other nodes but immediately fulfilled locally.
- **Receiving q .** Again, n checks if $r \subseteq R_n$. If so, it could return r to the node that posted q . As well, n would check if $q \subseteq Q_n$. If so, n might elect to not forward q (as r will be contained in the set of responses to Q_n).
- **Receiving a response $r' \notin R_n$.** Here, n checks if any queries in Q_n now have complete responses from $(r' \cup R_n)$. Any such responses could then be returned to their corresponding query sources.

In dynamic topologies, factors like temporary partitions and packet drops and expirations make it difficult to know whether it is better to wait for the responses to Q_n or immediately forward $q \subseteq Q_n$. Accordingly, in our model query filtering occurs only at the query's source and via subsumption. That is, either node n posting query q has $r \subseteq R_n$ (so q can be locally fulfilled) or does not (so n transmits q in whole). This approach also assists predictive caching, as the replication of responses continuously refreshes the caches of all nodes.

Checking for subsumption only at query sources also reduces the computational burden. Checking only when n posts a new query q is much less onerous than checks after every receipt of a query or response. We further reduce processing time by adopting the probabilistic method from [24] given in Algorithm 1. Although fast, this method can falsely indicate subsumption relationships; therefore, it is best adopted when queries seek a representative sample of data in an area rather than require complete, exact matches.

Algorithm 1. Checks if the response to a query q is subsumed in an array $r[]$ of n responses

Inputs: SampleSize s (the number of points to be checked), Query q , Response array $r[]$

Output: Boolean variable *subsumed* (true if subsumption is present, false otherwise)

```
Point[s] p = new empty array, boolean subsumed = true
for (i = 0; i < s; i++) {
    do { p[i] = getRandomPoint(q); // select random point in query space
    } while (duplicate(p, p[i]); // repeat if p[i] already in Point[] array }
for (i = 0; i < s; i++) {
    boolean pointcheck = false; // tracks each point-checking iteration
    for (j = 0; j < n; j++) {
        if (inResponse(r[j], p[i]) { // is point p[i] in response r[j]?
            pointcheck = true;
            break; } // if r[j] contains point p[i], no need to check other responses}}
    if (!pointcheck) // if point p[i] is not in r[], the subsumption check has failed {
        subsumed = false;
        break; }}
return subsumed;
```

Fig. 3. Pseudo-java code for Algorithm 1

In this paper, we do not explicitly consider categorical variables. Although the basic approach of checking a node's response array would still apply, the random point selection would likely not apply. For example, suppose a query $q(X)$ seeks all vehicles of some type X ; then, the query space would effectively have only one point. The question, then, is if the set of matching data in the response array comprise a complete response to $q(X)$. Although not insuperable, this is a different problem from the range-based queries we consider here and one of our future research directions.

4.4 Theoretical Performance Gains

We now mathematically analyze how our two models change the depth of information search and dissemination for SBS and CACL. The basic SBS model allows flooding to proceed equally in all directions and, as shown in [25], causes flooding areas to be inversely related to the volume of same-size responses being forwarded. This fact allows estimating the performance gains under idealized conditions with the following simplifying assumptions.

Given a query q posted by a node n and seeking a matching response r :

- Exactly one copy of r exists in the network within a constant distance d of n .
- The response r will be delivered to n iff it exists within a constant distance Δ of n , where $\Delta < d$.
- Within the circle of radius d centered on n , the same response r is equally likely to be at any location.

For aggregation, we further assume a constant overlap ω (as defined in 4.2). Then, these assumptions, Δ can be estimated from the network-wide probability of response delivery p , where $p = \frac{\Delta^2}{d^2}$ and thus $\Delta = d\sqrt{p}$, for $0 \leq \Delta \leq d$. Higher overlap ω allows responses to be obtained from more distant nodes. Disregarding query packets, pairwise aggregation allows the flooding area A for each response can increase by a factor of $(1 + \omega)$ and Δ by $\sqrt{1 + \omega}$. If query packets are assumed to compose a constant fraction k ($0 < k < 1$) of network traffic, then pairwise aggregation increases A by a factor of $(1 + \omega)(1 - k)$ and Δ by $\sqrt{(1 + \omega)(1 - k)}$.

The adoption of different wireless standards will, of course, impact the parameters d , Δ , k , and ω . For example, using a higher data rate will directly increase d and Δ . Further, all else equal, having responses replicate farther will cache more varied response data at each node and increase ω . Network topology and density also impact those parameters; *e.g.*, k increases as it takes queries longer to find matching responses. Still, the relationships $p = \frac{\Delta^2}{d^2}$ and $A(\omega) = A(1 + \omega)(1 - k)$ are independent of such factors.

In contrast, subsumption-checking prevents queries from entering the network altogether. In effect, the network must fulfill fewer queries, in turn permitting each one a larger flooding area. This will be reflected in higher delivery (from increased search depth for queries) or faster response time (due to reduced competing network traffic). As with aggregation, the changes in flooding area and depth can be estimated from the improvement in delivery.

Network topology may prevent achieving the full theoretical gains. Traffic signals, for example, can inhibit routing by creating large gaps in vehicle flows. On the hand, if the location of a query's destination is known, high node densities can be exploited to establish more direct routes.

Limiting flooding areas for improving SBSB's throughput was explored in [26]. That approach would, however, tend to reduce the gains from aggregation and filtering, as nodes would process less varied response data.

Computation time is another concern. Ideally, any aggregation or filtering could be performed between broadcasts. Larger response packets and higher node density increase the allowable time, while queue length increases the required computation time. If such computations are too onerous,

an alternative would be to assess overlap for a random sample of the queue. We note the average times available for such computations in regard to our two simulation scenarios in section 5.

5 Simulations and Performance Analysis

Our simulations are conducted using the JiST/SWANS platform [2], which is a widely-used Java-based discrete event network simulator. Each of our simulation scenarios considers a set of 600 vehicular nodes moving along roads contained within a 1000m square over 40s of simulated time. Vehicles have a maximum speed of 13.4m/s (corresponding to 30mph, a common urban speed limit in the United States). We used the STRAW mobility module [8], a micro-simulator that considers real-world variables such as acceleration and stopping distances.

The goal of our simulations is to observe how the performances of redundancy reduction methods are affected by the prevalence of redundant data and volume of network traffic. In order to minimize the effect of random factors (such as node density variation) across simulation runs, we adopted a uniform density model with the following custom road layout, as shown in Fig. 4.

The layout is twenty 900m road segments, spaced every 100m, in a rectangular grid. Vehicles are initially distributed uniformly and are randomly assigned to travel clockwise or counterclockwise in a 500m square. For example, a node starting at point *A* (Fig. 4) would follow either the circuit (indicated in yellow) *A-B-C-D-A* or *A-D-C-B-A*. In effect, every intersection will be the start and end of one route in each direction, clockwise and counterclockwise. Further, each such pair of routes is assigned 1% of the total vehicular traffic, half in each direction.

In order to simulate realistic travel behaviors of vehicles moving toward specific destinations, each simulation run models 40s of real time. This shortness ensures that vehicles will complete less than half a circuit during a simulation run. Although our mobility model is not wholly realistic, all vehicles are effectively proceeding directly to a destination. Moreover, the uniform traffic distribution reaches equilibrium quickly; longer simulation runs do not materially affect the network performance results. In any event, real data for large groups of vehicles at this level of detail is not readily available to researchers [31].

Fig. 4. Road layout.

Because of the relatively high node density in our simulations, we use a shorter transmission range than the 300m to 1000m typical of VANET research. For dense urban environments (as opposed to highways), shorter transmission ranges simply enable greater network throughput without great risk of connectivity gaps. The 802.11p (5.9GHz) wireless standard from [28] gives a data rate of 12Mbps and mean transmission range of 50m with standard deviation 4m. Although JiST/SWANS does not yet simulate 802.11p, its 802.11b has a similar 11Mbps data rate and we set the mean transmission range to 45m to approximate the same network throughput, the shorter range of 802.11b here compensating for its lower data rate.

We include two scenarios defined by the network-wide query posting rate: a low network traffic one with 15 queries posted per second (in 5.1) and a high network traffic one with 30 per second (in 5.2). To scale network load linearly, the query variety is proportional to the posting rate – 400 different query specifications are possible in 5.1 and 800 in 5.2. For each query, only one node has the matching response; however, we do not assume this is known, so even after being matched, queries (and any appended responses) continue to replicate throughout the network.

Mapping of queries and responses is abstracted; our concern is the prevalence of data redundancy rather than its causes. Queries and their matching responses are mapped to a square search space with length L ; this corresponds to the prevalence of redundant response data rather than any particular distance within our simulations. Within each scenario, L may vary from 50 to 150 units.

Each query q is a rectangle with length $x(q)$ and width $y(q)$. The dimensions x and y vary from 5 to 15 units and are taken separately from a uniform distribution. Each unit represents 10 bytes of data; a query's range may contain from 25 to 225 units and the corresponding response from 250 to 2250 bytes.

Each data point represents the average results from ten simulation runs of 40s each, with crossbars indicating one standard deviation above and below the average. We present results for *delivery* (the percentage of queries that receive a matching response before expiry), *response time* (the average time between when a query is posted and the posting node receives the matching response), *response packets* (total broadcasts of response packets, with aggregated responses counting as one broadcast), and *query packets* (total broadcasts of unmatched queries). Results for packet delivery and response time are taken only from queries posted during the interval (10s, 30s). This gives 10s for the network to reach equilibrium traffic and provides a complete lifetime to all tracked queries. The four models considered are:

- Baseline (B): No aggregation or subsumption-based filtering. Since this model is invariant with respect to L , only one batch of simulations was run for each scenario and results appear as straight lines.
- Aggregation (A): Aggregation is used but not subsumption-based filtering.
- Subsumption (S): Subsumption-based filtering is used but not aggregation.
- Aggregation and Subsumption (AS): Both aggregation and subsumption-based filtering are used.

Summary of Results:

- The aggregation model provides some improvement in delivery at low values of L (*i.e.*, when redundancy is most common).
- The subsumption model provides substantial improvements in both delivery and response time over the entire tested range of L .
- For the aggregation and subsumption model, no statistically significant difference in delivery is observed from subsumption applied alone.

5.1 Low Network Traffic

In this scenario, the network query posting rate was 15 per second, corresponding to one query per vehicle every 40s. For each posted query, one node within 500m of its source holds the matching response. Results for delivery are given in Fig. 5, below. The B model provides delivery of about 86%. At the low end of query range L , the A model improves delivery by about 5%. However, this improvement rapidly dissipates as L increases. For testing the null hypothesis that the true mean values of the baseline and the other models are the same, p-values are given in Table 2. The S and

AS models both show a large and statistically significant improvement over the baseline throughout the entire range of tested L values. They further are very similar, even at low L values, suggesting the S model alone encapsulates the potential performance gains of aggregation.

L	Aggregation	Aggregation and Subsumption	Subsumption
50	0.00206	9.1E-06	9.0E-06
70	0.00307	9.9E-06	2.2E-06
90	0.0306	3.0E-06	1.0E-04
110	0.110	9.3E-06	2.7E-05
130	0.0796	8.3E-06	8.3E-05
150	0.149	3.3E-05	2.7E-05

Table 2. Two-tailed p-values for delivery against baseline, low traffic.

Response times are given in Fig. 6. The A model is very similar to the baseline, although slightly slower. This arises from the cumulative effect of small delays in forwarding response packets; with the larger aggregated packets, the first transmission of any response packet by a given node tends to occur later. The S and AS models, on the other hand, greatly reduce response time because many queries can be locally fulfilled using locally available data.

Fig. 5. Delivery, low traffic

Fig. 6. Response time, low traffic

Fig. 7. Query packets, low traffic

Fig. 8. Response packets, low traffic

Fig. 9. Aggregation, low traffic

Packet counts are given for queries and responses in Figs. 7 and 8, respectively. These counts only consider transmissions during the middle 20s of each simulation run. Some counter-intuitive effects are observed. For the A model, fewer response packets are broadcast than in the baseline but more query packets are. This occurs because aggregation allows all response packets to be transmitted sooner overall. Because SBS with CACL selects packets according to their prior broadcasts, this causes queries to be broadcast sooner as well. Also, note that for the AS model, both response and query packet counts are lower than in the baseline. The cause is less favorable aggregation options. Overlap tends to be lower, so response packets tend to be larger than in the A model; this crowds out transmissions of query packets.

Observed aggregation *rates* (the fraction of response packet transmissions that are in fact aggregated) and overlaps (the magnitude ω) are given in Fig. 9 for models A and AS. Models B and S do not use aggregation and are omitted. Both the rates and overlaps decrease as the query range L increases (and the rate and overlap curves converge). Likewise, the rate is lower for model AS at low values of L because subsumption relationships are fairly common. Note that the product of rate and overlap indicates the expected throughput increase; even allowing for more collisions, the delivery increases due to aggregation fall far short.

For example, at $L = 50$, the product is about 0.3. Using our performance gains model from 4.4, for the baseline responses should be obtainable from about 86% of the area around a given query source (*i.e.*, the circle with radius 500m). For the A model, then, responses should be obtainable from an area of $(1.3)(0.86) = 1.118$ times as large. Thus, delivery should be virtually 100%. This is

not simply due to repeat selection of the same aggregation candidates, as evidenced by the unimpressive gains both at high and low rates. Instead, the main cause is interruptions in vehicle flows from traffic signals, which prevents response forwarding to new nodes and so induces repeated broadcasts of the same aggregates to nodes already having them.

5.2 High Network Traffic

In this section, we observe how the four models perform under a heavier network load. Queries are posted at a network rate of 30 per second, corresponding to one per vehicle every 20s. As in 5.1, again one node within 500m of each query source has the matching response. The higher posting rate reduces the flooding area for each query, which we expect to reduce delivery for the baseline model. The higher posting rate will also make nodes process a greater variety of responses, potentially allowing more opportunities for aggregation and subsumption.

Results for delivery are given in Fig. 10. Although the baseline model achieves only about 65% delivery, the pattern resembles Fig. 5. Again, the A model increases delivery by about 5% at $L = 50$ and rapidly converges to the baseline as L increases. This is because the set of secondary packet candidates at each node is essentially the same as in 5.1. Since node density and response size distribution are the same, each node's minimum forwarding utility u_{min} would be the same. For testing the null hypothesis that the true mean values of the baseline and the other models are the same, p-values are given in Table 3. As we observed in our low network traffic scenario, the S and AS models both show a statistically significant improvement over the baseline throughout the entire range of tested L values. The similarity of the S and AS model results again shows that the S model encapsulates the potential performance gains of aggregation.

L	Aggregation	Aggregation and Subsumption	Subsumption
50	0.000247	2.6E-11	3.6E-10
70	0.0185	6.1E-12	4.3E-12
90	0.176	1.9E-10	2.7E-10
110	0.0949	2.4E-07	6.9E-06
130	0.266	5.0E-07	9.3E-06
150	0.534	2.2E-05	3.2E-06

Table 3. Two-tailed p-values for delivery against baseline, high traffic.

The S and AS models again provide almost 100% delivery at $L = 50$ and almost a 10% gain over the baseline at the highest tested values of L . This is partly due to the ability to reuse local data. That is, while pairwise aggregation at best can increase throughput by a factor of two, subsumption can locally answer any number of queries seeking similar data. Still, as the query range increases, S and AS converge to the baseline's performance. No material differences were observed between the S and AS models.

Response time (Fig. 11) also resembles the results from 5.1, although somewhat slower for all four models. The curve for the baseline is almost perfectly aligned with that of the A model. Because of the higher query posting rate, a greater proportion of network traffic is devoted to forwarding them. Note that all four models show substantially higher query packet counts here (Fig. 12) than they did in 5.1. The response packet counts (Fig. 13) are much nearer to their 5.1 values because response packets are much larger than the unmatched queries.

However, we do observe slightly higher aggregation rates and overlaps in both the S and AS models (Fig. 14) because CACL considers the effect of aggregation when estimating a node's future transmission capacity. That is, when aggregation is more extensive, nodes can transmit more responses per unit time, decreasing their u_{min} and letting them select secondary packets from a larger set of candidates. This effect is not, however, sufficient here to achieve any noticeable gains in delivery or response time (comparing A to B or AS to S).

Fig. 10. Delivery, high traffic

Fig. 11. Response time, high traffic

Fig. 12. Query packets, high traffic

Fig. 13. Response packets, high traffic

Fig. 14. Aggregation, high traffic

6 Conclusion

Within current technological capabilities and wireless standards, long transmission distances are poorly suited for the dense vehicle populations commonly observed in cities. This is especially true when vehicles seek information regarding vehicles and traffic flows in their own immediate vicinities. Short transmission distances avoid congesting the network with information beyond the area in which it would be of wide interest. Yet, the data processing capabilities of mobile devices continue to grow exponentially. Accordingly, it is worthwhile to consider not only the potential of shorter transmission distances but also how data might be managed in order to reduce the network's bandwidth requirements. We have compared two localized models for reducing redundant data transmissions in such dense vehicular environments; to the best of our knowledge, this comparison has not been previously performed for VANET scenarios.

We observed that pairwise response aggregation can provide material gains in response delivery when the query range is very limited. However, we also observed that vehicular mobility limited the gains to a fraction of their theoretical potential. In contrast, query filtering by subsumption gave much greater performance improvements, both in delivery and response time. The potential cost is that locally obtained responses may not be current and some data points may be missed entirely. While our research should not be taken as a blanket condemnation of aggregation, it strongly suggests that filtering is much the better option for flooding-based applications.

Our future work will develop this query filtering model to provide more current response data. This will entail a probabilistic approach to refreshing data caches. Individual data elements will be transmitted according to how rapidly they change and the intensity of the demand for them. By applying a ranking function to such data and synthesizing response packets that may fulfill the missing parts of many queries, we expect to further improve the performance of our subsumption-based query filtering model.

7 REFERENCES

1. Bai, F. and Krishnamachari, B., "Spatio-temporal variations of vehicle traffic in vanets: facts and implications," in *Proceedings of ACM VANET '09*, 43 – 52, 2009.
2. Barr, R., Haas, Z., and van Renesse, R., "JiST: An efficient approach to simulation using virtual machines," *Software Practice and Experience*, vol. 35, no. 6, 539 – 576, 2005.
3. Bononi, L. and di Felice, M., "A cross layered MAC and clustering scheme for efficient broadcast in VANETs," in *Proceedings of MASS 2007*, 1- 8, 2007.
4. Campolo, C., Molinaro, A., and Vinel, A., "Understanding the Performance of Short-lived Control Broadcast Packets in 802.11p/WAVE Vehicular Networks," *2011 IEEE Vehicular Networking Conference*, Amsterdam, The Netherlands, 2011.
5. Carzaniga, A., Rosenblum, D., and Wolf, A., "Design and evaluation of a wide-area notification service," *ACM Transactions on Computer Systems*, vol. 19, no. 3, 332 – 383, 2001.
6. Chandramouli, B. and Yand, J., "End-to-End Support for Joins in Large-Scale Publish/Subscribe Systems," in *Proceedings of PVLDB '08*, Auckland, New Zealand, 434 – 450, 2008.
7. Chen, R., Jin, W., and Regan, A., "Broadcasting Safety Information in Vehicular Networks: Issues and Approaches," *IEEE Network Magazine*, vol. 24, no. 1, 20 – 25, 2010.
8. Choffnes, D. and Bustamante, F., "An Integrated Mobility and Traffic Model for Vehicular Wireless Networks," in *Proceedings of the 2nd ACM International Workshop on Vehicular Ad-hoc Networks*, 2005.
9. Defude, B., Delot, T., Ilarri, S., Zechinelli-Martini, J., and Cenerario, N., "Data aggregation in VANETs: the VESPA approach," *First Int'l Workshop on Mobile and Ubiquitous Systems: Networks and Services (MOBIQUITOUS'08)*, Dublin, Ireland, 2008
10. Dietzel, S., Bako, B., Schoch, E., and Kargl, F., "A fuzzy logic based approach for structure-free aggregation in vehicular ad hoc networks," *6th ACM International Workshop on Vehicular Internetworking*, Beijing, China, 79 – 88, 2009.
11. Dikaiakos, M., Iqbal, S., Nadeem, T., and Iftode, L., "VITP: An information transfer protocol for vehicular computing," in *Proceedings of the 2nd ACM International Workshop on Vehicular Ad-hoc Networks*, 2005.
12. Ghafoor, K., Bakar, K., van Eenennaam, M., Khokhar, R., and Gonzalez, A., "A fuzzy logic approach to beaconing for vehicular ad hoc networks," *International Journal of Telecommunication Systems*, vol. 53, no. 4, 330 – 342, 2011.
13. Gupta, S., Joshi, A., Santiago, J., and Patwardhan, A., "Query Distribution Estimation and Predictive Caching in Mobile Ad Hoc Networks," *International Workshop on Data Engineering for Wireless and Mobile Access*, Vancouver, Canada, 24 – 30, 2008.
14. Hara, T., "Effective Replica Allocation in Ad Hoc Networks for Improving Data Accessibility," in *Proceedings of IEEE INFOCOM*, 2001.
15. Ibrahim, K., "Data Aggregation and Dissemination in Ad-Hoc Networks," Doctoral Thesis, Old Dominion University, Norfolk, Virginia, 2011.

16. Jurca, O., Michel, S., Herrmann, A., and Aberer, K., "Continuous Query Evaluation over Distributed Sensor Networks," in *Proceedings of IEEE ICDE 2010*, Long Beach, California, 912 – 923, 2010.
17. Jurca, O., Michel, S., Herrmann, A., and Aberer, K., "Query Driven Operator Placement for Complex Event Detection over Data Streams," in *Proceedings of EuroSSC*, Zürich, Switzerland, 2008.
18. Jurca, O., "A Publish/Subscribe Approach to Processing Continuous Queries over Sensor Data Streams," Doctoral Thesis, EPFL, Lausanne, Switzerland, 2010.
19. Legendre, F., Hossman, T., Sutton, F., and Plattner, B., "30 Years of Wireless Ad Hoc Networking Research: What about Humanitarian and Disaster Relief Situations? What Are We Still Missing?" *International Conference on Wireless Technologies for Humanitarian Relief*, Kerala, India, 2011.
20. Lim, Y., Ahn, S., and Cho, K., "Abiding Geocast for Commercial Ad Dissemination in the Vehicular Ad Hoc Network," *IEEE Conference on Consumer Electronics*, Berlin, Germany, 2011.
21. Lochert, C., Scheurmann, B., and Mauve, M., "A probabilistic method for cooperative hierarchical aggregation of data in VANETs," *Ad Hoc Networks*, vol. 8, no. 5, 518 – 530, 2010.
22. Lundquist, D. and Ouksel, A., "A Context-Aware Cross-Layer Broadcast Model for Ad Hoc Networks: Analysis of Multi-Hop Routing," *Procedia Computer Science*, vol. 10, 766 – 774, 2012.
23. Nair, R., Soh, B., Chilamkurti, N., and Park, J., "Structure-free message aggregation and routing in traffic information system [SMART]," *Journal of Network and Computer Applications*, vol. 36, no. 3, 2013.
24. Ouksel, A., Jurca, O., Podnar, I., and Aberer, K., "Efficient probabilistic subsumption checking for content-based publish/subscribe systems," *ACM/IFIP/USENIX 7th Int'l Middleware Conference*, Melbourne, Australia, 2006.
25. Ouksel, A. and Lundquist, D., "Demand-driven publish/subscribe in mobile environments. Wireless Networks," vol. 16, no. 8, 2237 – 2261, 2010.
26. Ouksel, A. and Lundquist, D., "A Context-Aware Cross-Layer Broadcast Model for Ad Hoc Networks," *Journal on Personal and Ubiquitous Systems*, vol. 18, no. 4, 851 – 864, 2014.
27. Rajagopalan, R. and Varshney, P., "Data aggregation techniques in sensor networks: a survey. IEEE Communications Survey Tutorial," vol. 8, no. 4, 48 – 63, 2006.
28. Sai, S., Oshida, T., Onishi, R., Yoshioka, A., and Tanaka, H., "Comparisons of Non-Line-of-Sight Inter-Vehicle Communications in the Urban Environment Between 5.9GHz and 700MHz Bands," *2012 IEEE Vehicular Networking Conference*, Seoul, South Korea, 2012.
29. Srivastava, D., "Subsumption and indexing in constraint query languages with linear arithmetic constraints," *Annals of Mathematics and Artificial Intelligence*, vol. 8, 315 – 343, 1992.
30. Uddin, M., Wang, H., Saremi, F., Qi, J., Abdelzaher, T., and Huang, T., "PhotoNet: A Similarity-aware Picture Delivery Service for Situation Awareness," in *Proceedings of IEEE RTSS 2011*, Vienna, Austria, 2011.
31. Uppoor, S. and Fiore, M., "Large-Scale Urban Vehicular Mobility for Networking Research," *2011 IEEE Vehicular Networking Conference*, Amsterdam, The Netherlands, 2011.
32. van Eenennaam, B., Karagiannis, G., and Heijenk, G., "Towards Scalable Beaconing in VANETs," *4th ERCIM Workshop on eMobility*, Luleå, Sweden, 103 – 108, 2010.
33. Vinel, A., "3GPP LTE versus IEEE 802.11 p/WAVE: Which technology is able to support cooperative vehicular safety applications?" *IEEE Wireless Communications Letters*, vol. 1, no. 2, 125-128, 2012.
34. Vinel, A., Belyaev, E., Egiastian, K., and Koucheryavy, Y., "An overtaking assistance system based on joint beaconing and real-time video transmission," *IEEE Transactions on Vehicular Technology*, vol. 61, no. 5, 2319-2329, 2012.
35. Wisitpongphan, N., Tonguz, Ö., Parikh, J., Mudalige, P., Bai, F., and Sadekar, V., "Broadcast Storm Mitigation Techniques in Vehicular Ad Hoc Networks," *IEEE Wireless Communications*, vol. 14, no. 6, 84 – 94, 2007.
36. Yang, L., Guo, J., and Wu, Y., "Channel Adaptive One Hop Broadcasting for VANETs," *11th International IEEE Conference on Intelligent Transportation Systems*, Beijing, China, 369 – 374, 2008.

37. Yu, B., Gong, J., and Xu, C., “Catch-up: a data aggregation scheme for vanets,” in *Proceedings of 5th ACM International Workshop on Vehicular Ad Hoc Networking*, San Francisco, 49 – 57, 2008.