Congestion Control for Epidemic Routing in Opportunistic Networks

Raphael Bialon Kalman Graffi
Department of Computer Science, University of Düsseldorf, Germany
{bialon, graffi}@hhu.de

***Abstract*—Flooding-based routing protocols in opportunistic networks like Epidemic provide a simple routing algorithm. While delivery probability is high and delivery latency is low, the generated overhead of distributing every message to each node generates unnecessary load in the network, especially in areas with higher node density and areas with network congestion. Several approaches try to control congestion by utilizing additionally transferred information to coordinate packet transmissions. Due to the nature of an opportunistic network, solutions designed to use a central or global oracle are seldom feasible. Our solution contributes a per-message probability-based decision for congestion control using ephemeral information of a nodes' surroundings. Simulations conducted using the ONE simulator show, that after a certain node density resulting in a congested network area, overhead ratio decreases and delivery probability significantly increases compared to Epidemic routing. By testing scenarios with different movement models, we also foreclose algorithm efficiency based on node interactions.**

***Index Terms*—Opportunistic networks, Epidemic, flooding, congestion, ONE**

I. Introduction

As a subclass of *Delay Tolerant Networks* (DTNs), opportunistic networks take advantage of device mobility to distribute and route data through the network. Participants in an opportunistic network are mobile and wirelessly connected nodes. They behave highly dynamic regarding their movement and (dis-)connections. A predetermined and complete route through the network can neither be computed nor used, because the network, its fragments, and participants can change any moment. To route messages through an opportunistic network, a technique called Store-and-Forward [1] is used. Nodes exchange messages on mutual encounters and store new messages in their buffer. When another node connects, appropriate messages will be forwarded to it.

This technique allows for message routing in opportunistic networks, for which a multitude of routing protocols exist [2]. From routing protocols utilizing meta data exchanged with other nodes and establishing assumptions and predictions on future node behavior, to routing protocols relying completely on randomness, those distinct approaches all try to achieve a high probability of message delivery.

One category, the category of flooding-based routing protocols, aims at a high delivery probability without the need for any routing meta data exchange. Only data messages will be transferred. Epidemic [3] is a well-known flooding-based routing protocol. Its algorithm is simple, allowing an easy and widespread usage: Upon node encounter all messages, except those that have already been transferred to the other node, are sent. Thus, messages reach their destination on the fastest possible way as all potential ways through the network are tried, but incur a high overhead [4]. This high overhead is the result of the high number of messages copies, as every encountered node carries one more copy of the message and then continues flooding the network. As a result of the high overhead in the network in general and on each node, *network congestion* occurs.

A definition for network congestion is a "[...] node carrying so much data that its quality of service (QoS) deteriorates. Typical effects include queueing delay, transmission delay, packet loss or the blocking of new connections." [5]. Nodes can cause a situation of congestion for themselves and surrounding nodes. For others, because transmissions of large amounts of messages takes up time, which cannot be used by other nearby nodes, and also for themselves, because forwarding of messages to their destinations may not occur if blocked by other preceding transmissions. Over time, node buffers are filled up further with messages stored for future forwardings and network congestion also accumulates, leading to an increased deterioration of the network. As each nodes transmissions possibly contribute to network congestion, the transmission behavior of each node has to be regulated individually.

The data needed for calculations of transmission behavior to employ Congestion Control can be taken from many different sources. The fastest and most reliable — in terms of trust — possible way to accumulate this data is by taking measurements from the device itself and passively gathered information of its surroundings. In contrast to already established methods, our strategies utilize only locally available data, e.g. by collecting local statistics without the need of explicit data exchange with other nodes. Therefore, calculations are done solely on the device itself which is preferable in the light of recent stance towards data privacy as no potential data could be leaked [6].

As the contribution of this paper, we introduce two Congestion Control strategies aiming at reducing the overhead for Epidemic routing in congested environments while maintaining a high delivery ratio. We motivate which data to use and how to combine selected measurements into a mathematical decision making process. Our Congestion Control strategies then reduce the amount of transmissions as compared with Epidemic depending on the nodes immediate surroundings. We suggest that reduced transmissions in an environment with

a high node density, meaning a high number of neighboring nodes for the examined node, lead to an overall lower overhead ratio. It also enables other nodes to participate in transmissions whereas otherwise congestion caused by too many nodes trying to transmit at the same time would have prevented them from doing so. Other data is taken into account to ensure a steady amount of transmissions over time, preventing single nodes from taking an unfair share. To also improve delivery ratio, we fit messages that have not yet travelled far with a higher preference in the transmission decision making process.

As congestion is associated with node count and density, we utilize different node counts in our evaluation to represent varying degrees of congestion. With Epidemic routing, this leads to higher overhead ratios but also improves delivery probability as more nodes can partake in the process of message delivery, increasing the probability of one forwarding node transmitting the message to its destination. This permits our Congestion Control strategies to assess their impact on Congestion Control and to validate our previously stated goal. The results show that our Congestion Control strategies improve general network behavior while maintaining a behavior similar to Epidemic in situations without network congestion.

The remainder of this paper is structured as follows: First, we position our work with regards to existing publications. Then, we introduce the Congestion Control strategies, their underlying data, and how they are coupled together. By exhibiting results of our strategies' calculations we show the impact of our algorithm on Congestion Control. Simulation scenarios to evaluate our methods in and metrics used for the evaluation are then defined. Finally, we draw a conclusion from our work and discuss our contribution.

II. Related Work

When using a flooding-based routing protocol like Epidemic [3], Congestion Control is one of the key features to prevent overloading the network. Many other publications offer different approaches to achieve different kinds of Congestion Control, all using different approaches. In the following, we give a short overview on the existing work and describe the significant differences compared to our protocol.

Dvir et al. [7] present a congestion avoidance strategy which takes a nodes' energy usage into account. They focus on a low-energy solution to improve delivery probability by trying to predict the amount of future transmissions of a given packet. In difference to our approach, this algorithm relies on the exchange of information before forwarding messages.

Radenkovic et al. [8] propose a hybrid approach that combines several metrics from different layers, including social interactions, packet popularity and others, to offload congestion onto different parts of the opportunistic network. The computation is complex and uses heuristics in addition to keeping a history of node encounter information exchange.

Ippisch et al. [9] introduce a Markov-Chain based computation of a packets' optimal path through an opportunistic network. They rely on global information such as the number of nodes having already received a particular packet through a differential equation-based approximation. Our approach only takes into account directly measurable, ephemeral information of a nodes' surroundings. Similar to this, they propose a different strategy based on the same Markov Chain in [10]. While this aims at providing reduced message congestion, local network information is not taken into account, only message information. Both papers adopt observations gained from the Markov Chain onto routing algorithms, but leave out environmental influence on general network behavior.

Yun et al. [11] provide the AFNER congestion control strategy with tests conducted using the ns-2 simulator. While relying on information exchange before transmitting packets, they also take global information on the network state into account. This information may not always be available and due to the dynamic structure of an opportunistic network, it might also not always be up-to-date.

Lo et al. [12] focuses on an extension for quota-based routing protocols. As Epidemic is a flooding-based routing protocol, their findings cannot be directly applied onto it. Also, the exchange of meta-data information is necessary before any packet exchange can occur.

Summarizing, various efforts have already been made to reduce congestion for Epidemic routing. In contrast to existing publications, our congestion control strategies focus on privacy-oriented locally available data only, giving nodes the option of independent decisions.

III. Strategies for Congestion Control

Due to the nature of opportunistic networks, message routing cannot take advantage of predetermined paths and works on a best-effort basis. Flooding-based routing protocols like Epidemic attempt to utilize every possible way a message can travel through the network, possibly leading to a successful delivery. As Epidemic tries all possible ways for a message to travel the network, this also includes the optimal way. We argue that this high amount of message transfers is fine for network environments with sufficient network performance, induced through e.g. low node density or infrequent message generation, but it has to be reduced for environments with degraded network performance. Therefore, our Congestion Control strategies adapt to the network environments.

While many established methods for Congestion Control rely on device-external information either sent as additional information ahead of packet transmission or by querying a (global) oracle, our algorithm obtains the information it processes directly from a devices' known data. In this section we introduce the information and their relationship to our idea of Congestion Control, as well as defining the functions used by our algorithm to generate a probability for Congestion Control.

A. General idea

A regular node participating in an opportunistic network learns some information about its surroundings by actively or passively partaking in the network and by querying information from packet buffers and local transmission history.

While Epidemic per se floods a network irregardless of its current status, we want to reduce the number of transmissions every single node in a congested area emits. Each packet transmission is then assessed by consulting the transmission probability of a node and the packet's current state on its way through the opportunistic network. As the values used in our algorithm are quite unsteady due to the dynamic nature of opportunistic networks, they should only be used to generate an ephemeral status of a nodes' vicinity. Therefore, these values should be updated regularly and only the most recent one of each have to be taken into account for calculating a transmission probability.

B. Information used

Though only a limited set of information is available from a node's point of view (without having external information), many combinations can be extracted from those as can be seen in [13], for example. We focus on a subset of available information, and define and motivate our selection in the following. We use x to denote a node and m for a message.

Count of neighbors: $N(x)$. The first and most obvious property we consider is the count of neighbors. A node can be aware of its immediate surroundings by noting the number of nodes it can communicate with. In addition, it can observe communications of others and learning of new neighbors it previously wasn't aware of because no communication between both had taken place.

Count of messages in buffer: $M_b(x)$. As one indicator of how many messages we want to forward and thus introduce as new transmissions in the network, the count of enqueued messages in the buffer is one good choice. If a nodes' buffer contains many messages, in flooding-based routing it is also expected to emit many messages into the network. Reducing this number of transmissions frees resources which can then be used by other nodes transmissions.

Transmission statistics: $M_a(x)$ and $M_c(x)$. A node can keep track on the count of messages sent per unit of time, thereby generating statistics on its transmission behavior over time. This information acts as a feedback, allowing a node to compare its calculated transmission behavior to the actual one. We measure the average number of messages sent per minute as $M_a(x)$ and the number of messages already sent during the current minute as $M_c(x)$. $M_a(x)$ is updated and $M_c(x)$ is reset every minute.

Count of potential messages: $M_p(x)$. While the count of neighbors and the count of messages in buffer are valid guides on transmission behavior on their own, they can be combined into the count of potential messages, resembling the maximum number of transmissions a node has to emit in order to forward every message in its buffer to every neighbor who has not already received it. Message forwarding is generally done via Unicast messages to the designated receiver only, also allowing, as an example, for individual transport encryption. Broadcast messages would require too many neighbors to listen on transmissions. Defining the count of interested neighbors as the count of neighbors, who have not yet had this message forwarded to, and the count of forwardable messages as the count of messages a neighbor has not yet had forwarded, we can describe its calculation as follows: This count is increased by the amount of interested neighbors every time a new messages is received or created, and when a new neighbor appears by the amount of forwardable messages. It is decreased by one whenever a transmission succeeds and it is also decreased by the amount of forwardable messages upon the node, to whom those message could have been forwarded to, leaving the neighborhood (e.g. through a timeout or beaconing). When a message is dropped, the amount of interested neighbors is subtracted from the current count of potential messages. This count gives a value of the load a node is going to burden on the network from its current point of view.

Packet hop count: $h(m)$. As a measure of an individual packets importance for Congestion Control, hop count is used to identify how far a packet has traveled through the network. A packet with a higher hop count is expected to have traveled for some time throughout the network, but most importantly, it is also expected to be available in more node buffers than a packet with lower hop count. We use this value as an indicator of how important a packets' transmission is in addition to the general probability calculation of a nodes' transmissions.

Table I
VARIABLES, INPUT AND OUTPUT FUNCTIONS

Variables	
x	A node
m	A message
Input functions	
$N(x)$	Neighbor count
$M_b(x)$	Count of Messages in buffer
$M_a(x)$	Average count of messages sent per minute
$M_c(x)$	Count of messages already sent during current minute
$M_p(x)$	Count of potential messages
$h(m)$	Hop count of message
Output functions	
$P(x)$	Node sending probability
$F_1(x, m)$	Per-message fitness function variant 1
$F_2(x, m)$	Per-message fitness function variant 2

C. Derivation of Congestion Control Strategies

To utilize the information stated in the former section, we give a mathematical explanation of how these measures are used in order to form a probabilistic function. The definitions and measures we are going to use in our calculations are stated in table I. All functions return a result valid for the current time t, which is not explicitly named as a parameter to each function and measure to reduce complexity.

Note that all of these functions only return values equal to or greater than 0.

Using these functions, we can motivate the values v_i forming our probabilistic function:

$$v_1(x) = M_p(x)^{-1} \tag{1}$$

Equation (1) indicates the portion of one message to all messages that could potentially be transmitted. Using this measure, we linearly decrease a part of transmission probability, resulting in a more steadily transmission behavior across different amounts of buffer load.

$$v_2(x) = \min\left\{M_b(x), \frac{M_a(x)}{N(x)}\right\} \cdot M_b(x)^{-1} \tag{2}$$

The outcome of eq. (2) describes how the average transmissions per minute in relation to the current neighbor count influences its part of the transmission probability. It is capped by using the minimum of the amount of messages currently enqueued in buffer and the aforementioned relation. By using this cap, it offers a probabilistic value of 1 if $M_a(x) \geq M_b(x) \cdot N(x)$, meaning that the node was able to send at least as many messages on average per minute in the past as it wants to send at most right now. For all other cases, it offers a probabilistic value representing the relation of messages sent per minute on average to the current neighbor count, normalized by the current amount of messages enqueued in buffer.

$$v_3(x) = \min\left\{M_b(x), \|M_a(x) - M_c(x)\|\right\} \cdot M_b(x)^{-1} \tag{3}$$

The third part of our probabilistic function as shown in eq. (3) gives an insight on how the current nodes' transmissions perform related to its average transmissions per minute in the past. Again, using the minimum of the count of messages enqueued in buffer, and the distance between the past average amount of transmissions per minute and the amount of transmissions during the current minute, it offers a probabilistic outcome with the functions' value capped at 1 by using the amount of messages currently enqueued in buffer as a normalization, too. It enables a higher probability when a nodes' transmission behavior is off from its average and lowers the probability as the current transmission behavior reaches the past average.

With $0 \leq v_i \leq 1$, $\sum_{i=1}^{3} w_i = 1$ has to hold, resulting in our probability function for sending messages from a nodes' view in general:

$$P(x) = \sum_{i=1}^{3} w_i \cdot v_i(x) \tag{4}$$

We chose $w_1 = w_2 = w_3 = \frac{1}{3}$ as every value represents one aspect of a nodes behavior towards congestion and should be weighted equally.

For obvious reasons, $P(x)$ is also defined to return 0 if any of $N(X)$, $M_b(x)$ or $M_p(x)$ returns 0.

Now we can obtain a probability for sending a message on node x during time t. But assigning the same probability to each message a node wants to send does not take message information into account. It is necessary to also include message information into the sending decision to further reduce transmissions. Thus, we have to include at least one property of a message into our calculations to obtain a per-message decision choosing whether to send or not.

As described in section III-B, we chose a message's hop count as the property taken into account, as it gives some insight on how far the message has already traveled. Because we know that we want to reduce unnecessary traffic in congested environments, we favor messages having a low hop count to quickly distribute them into our environment. Messages that have already traveled far and have a higher hop count accordingly will receive a lower score for further forwarding. Because we can assume that a larger number of nodes has already received this message and are also forwarding it further, this allows us to reduce the amount of transmissions of packets.

Therefore, we adapt the general probability to an individual value for each message by applying a derivate of the messages' hop count to it. We introduce two of these methods, which cover about the same value range but follow a different gradient. With $h(m)$ being the hop count of a message m at implicit time t on node x, we use the following multiplicands:

$$2 \cdot (h(m) + 1)^{-1} \tag{5}$$

$$\log(2 + h(m))^{-1} \tag{6}$$

Both eqs. (5) and (6) return a multiplicative factor greater than 1 for messages with a hop count of 0. This only applies to messages created by the node itself, leading to locally generated messages being transmitted with an increased factor compared to others.

Because applying a multiplicative factor with possible values greater than 1 to our probabilistic function $P(x)$ results in a value possibly exceeding the maximum probability value of 1, we integrate them into two fitness functions. The results of those fitness functions shown in eqs. (7) and (8) are still applied in a probabilistic decision making process, but values greater than 1 are regarded as having the same value as 1. The following two fitness functions, representing our Congestion Control strategies, are applied in our simulations:

$$F_1(x, m) = 2 \cdot (h(m) + 1)^{-1} \cdot P(x) \tag{7}$$

$$F_2(x, m) = \log(2 + h(m))^{-1} \cdot P(x) \tag{8}$$

These will be referred to as variants 1 and 2.

Please note, that in our simulations *First In, First Out* (FIFO) is used as the buffer management strategy. Our Congestion Control strategies act on messages sorted by the buffer management strategy and thus do not prefer sending locally generated messages before others, but increase the probability of a locally generated message to be sent when compared with others. Therefore, the impact of the chosen buffer management strategy is reduced. We also conducted simulations with a random buffer management strategy, which showed results similar to FIFO.

IV. Evaluation

In this section we define simulations scenarios and metrics used for our evaluation, present results from simulation runs, and evaluate them by utilizing the stated metrics.

Table II
SCENARIO SETTINGS

Simulation duration	$43\,200\,\mathrm{s} = 12\,\mathrm{h}$
World size	$4500\,\mathrm{m} \times 3400\,\mathrm{m}$
Number of hosts	10, 50, 100, 150, 200, 250 and 300
Movement model	`ShortestPathMapBasedMovement` and `RandomWalk`
Movement speed	0.5 to $1.5\,\mathrm{m/s}$
Router	`CongestionControlEpidemicRouter` and `EpidemicRouter`
Interface type	`SimpleBroadcastInterface`
Interface speed	$1\,\mathrm{MB/s}$
Interface range	$100\,\mathrm{m}$
Buffer size	$1\,\mathrm{GB}$
Buffer sorting	FIFO
Message TTL	$300\,\mathrm{min} = 5\,\mathrm{h}$
Event interval	25 to 35 s
Event size	1 to 10 MB

A. Simulation scenario

We compute our simulation scenarios by utilizing the *Opportunistic Network Environment* (ONE) simulator [14]. It is an event-based simulator written in Java and offers a broad range of opportunistic network related protocol and interface implementations. Our implementation and corresponding scenario settings can be found on Github[1].

Simulation scenario settings used in our simulations can be found in table II. This table describes the general values used in every simulation scenario and also includes multiple values for one setting, which were used in combination with the other given values to produce the examined scenarios. For example, every router and movement model combination is tested in scenarios with 10, 50, 100, 150, 200, 250 and 300 hosts. Due to the modular design of ONE, scenarios can easily be built by only modifying the settings file.

Node speed is taken from the default scenario settings of ONE representing pedestrians. A buffer size of $1\,\mathrm{GB}$ resembles less than $1\,\%$ of latest device storage sizes. A larger buffer is important for flooding-based routing algorithms, as every incoming packet has to be temporally stored before it can be forwarded. As no selection of suitable packets per node is done before transmission, larger buffer size is preferred to store more packets than with other routing algorithms.

While message Time-to-Live (TTL) and the message sending (Event) interval are also taken from ONEs' default settings, message (Event) size is increased compared to those. The given range represents small multimedia files, such as pictures or short videos, being sent through the network.

The interface is configured with a speed of $1\,\mathrm{MB/s}$ to resemble WiFi usage, and a range of $100\,\mathrm{m}$. This range leads to many nodes possibly being in transmission range and provokes network congestion.

[1]https://github.com/hhucn/the-one-epidemic-cc

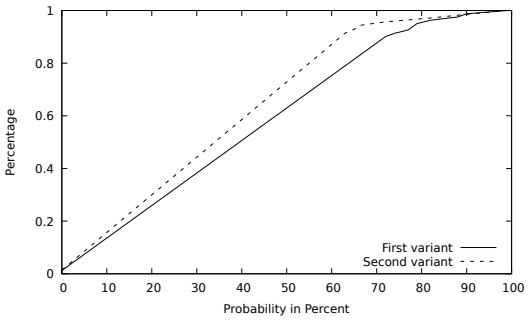


Figure 1. Cumulative distribution of probabilities for variants 1 and 2

We conduct all possible parameter combinations for all Congestion Control strategies and plain Epidemic using a map-based movement and random walk. This is done by employing `ShortestPathMapBasedMovement` and `RandomWalk` movement models in ONE, respectively. We expect the map-based movement scenarios to provide better results in general, as node encounters happen more often due to nodes following given routes. As an increased number of node encounters, and as a result an increased number of transmissions, happen, congestion becomes more likely to occur. Improvements in network behavior caused by our strategies are expected to be significant for these scenarios.

For the random walk scenario, we expect significantly fewer node encounters to happen. Nodes can move to any point within the simulation area and are therefore less likely to cross ways with another node. As a result, little to no congestion should occur. We selected these scenarios to show that our strategies do not impair network behavior when little to no congestion is present.

These scenarios allow the investigation of Epidemics' and our strategies behavior in environments with network congestion present and little to no congestion.

All simulation scenarios were run on a virtual machine using an Intel Xeon E31260L CPU and ECC RAM. As only seeded random number generators were used, all simulation runs are reproducible and should produce the same output on any machine supporting the Java Runtime Environment from version 8u181[2].

As a note for result reproduction, a simulation run for the Congestion Control Epidemic routing using the second version of our probability function for all seven given node counts took $9044.74\,\mathrm{s} = 2.51\,\mathrm{h}$ to complete.

B. Probability Distribution

To get an insight on how our probabilistic decision on packet sending affects transmission behavior, we show a cumulative distribution of the general probabilities P nodes have calculated and the results of our per-packet fitness functions F_1 and F_2 in figs. 1 and 2. Those values have been captured during the largest scenario utilizing 300 nodes and `ShortestPathMapBasedMovement`. Note the different

[2]https://www.oracle.com/technetwork/java/javase/downloads/index.html

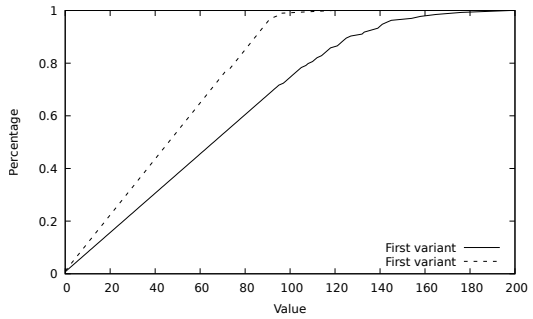


Figure 2. Cumulative distribution of per-packet values for variants 1 and 2

ranges for x values. The doubled value range on the x-axis is caused by our per-packet multiplication with the value derived from a packets' hop count. This leads to packets with a hop count of zero to have a fitness value twice as high as the general probability. With increasing hop count, the fitness value is decreased, enabling other packets with a lower hop count for a preferred transmission.

Although distorted by different maximum values, the slope of each graphs remains close to each other. This is expected, as the multiplicative supplement for per-packet calculation acts within a limited range and the creation of packets follows a normal distribution.

C. Metrics

For evaluation, we analyze our simulation runs by applying four metrics onto the data computed and provided by the ONE simulator. All of the following metrics are taken from ONEs' `MessageStatsReport`, an integrated reporting tool providing aggregated information after a simulation run has finished. The metrics, their calculation and significance for opportunistic network characteristics are stated in the following.

a) Delivery probability: The delivery probability is calculated as the ratio between packets created in the network and successfully delivered packets. It offers an insight on how well the network components handle a given workload, a higher value is better.

b) Overhead ratio: The overhead ratio is calculated as follows: With t as the number of total transmissions and d as the number of delivered packets, the overhead ratio then is $\frac{t-d}{d}$. This value shows how much overhead the routing protocol creates for the network, as each additional transmission could have instead been used for transmitting another, different packet. A lower value is preferred.

c) Average latency: The average latency is the average of all forwarding latencies. The forwarding latency is the time between packet creation and the first successful delivery. Again, a lower value is better.

d) Average hop count: The average hop count is calculated as the average of hop counts from all delivered packets. While a lower hop count not necessarily results in faster delivery, it is preferred to not fill the buffers of too many nodes with the same packet, resulting in less available buffer space for other packets.

Generalized, one can say that the favored outcome should include a high delivery probability, and low overhead ratio and average latency. A low average hop count can be preferred if the general network performance is not mitigated, but a higher hop count combined with higher delivery probability, for example, can also be preferred for a higher overall performance. Therefore, all of the stated metrics have to be examined in context to provide a valid evaluation for overall network characteristics.

D. Analysis

We group our simulation runs by the movement model used to allow for the comparison of different routing methods based on the same simulation scenario.

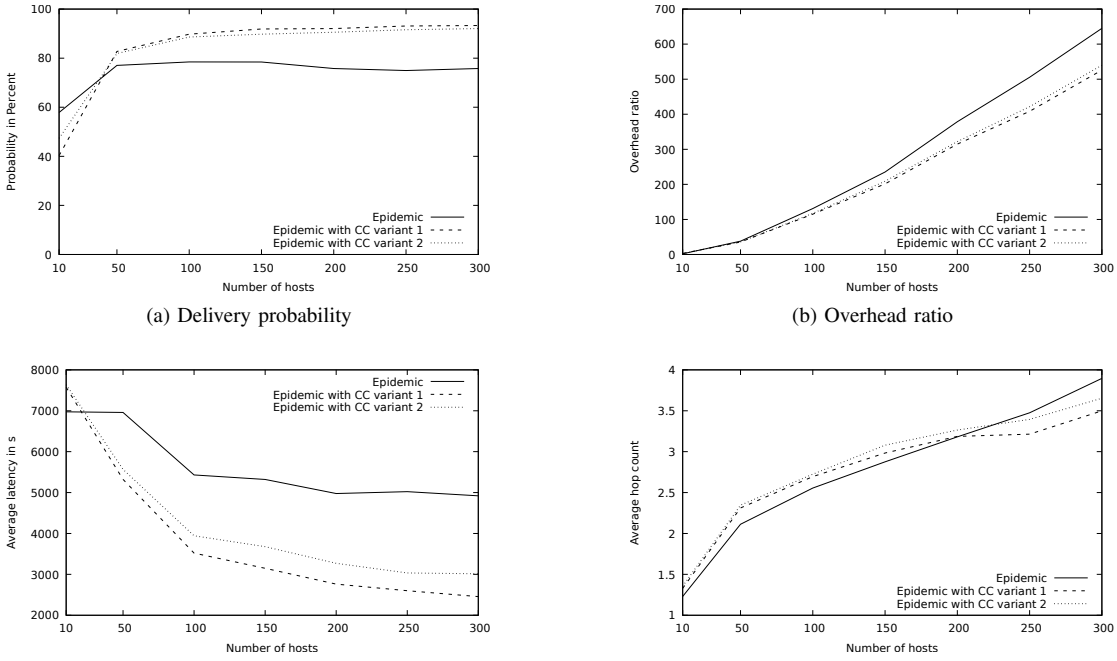
A flooding-based routing protocol like Epidemic already achieves a high message delivery probability. Contrary, due to the nature of opportunistic networks, a delivery probability of or really close to $100\,\%$ is extremely rare. Thus, network behavior has to be analyzed by taking all defined metrics into account. Improving one metric while impairing another does not improve network behavior in general, but instead only shifts the degradation to another property.

Therefore, in the following analysis, network behavior is analyzed by comparing all defined network metrics and not only showing single metric improvements.

We first focus on the map-based movement shown in fig. 3. While Epidemic routing has delivery probabilities always below $80\,\%$, both of out variants are able to achieve a delivery probability higher than $90\,\%$. The overhead generated by message replication for forwarding starts with slightly higher values than Epidemic routing for 10 nodes, but remains below Epidemics overhead with a gradually decreasing count. This clearly proves our initial goal as achieved, as we are able to generate fewer network transmissions while maintaining a higher delivery probability than Epidemic in congested environments.

The average hop count depicted in fig. 3d remains about the same as with Epidemic, a gradual improvement over the increasing amount of nodes can be observed. In the scenarios with 250 and 300 nodes, the average hop count is lower when utilizing Epidemic with our Congestion Control than Epidemic routing by itself. Although it only differs by a small factor, global node utilization per packet traveling through the network is reduced with an increasing number of participating nodes. Starting with scenarios containing more than 10 nodes, average latency using our Congestion Control mechanisms is lower than without, with finally halving average latency as compared to Epidemic when using the first variant of our algorithm.

For the map-based movement model, we can conclude that our extensions to Epidemic routing, by using a probabilistic function to decide if a transmission should take place or not, improves general network utilization. It also enables a superior use of network resources the more congested a scenario



(a) Delivery probability

(b) Overhead ratio

(c) Average latency

(d) Average hopcount

Figure 3. Metrics output on Epidemic routing with and without congestion control strategies for the map-based movement scenario

becomes, resulting in higher delivery probability and less load on individual nodes per packet as compared to Epidemic routing. Comparing both of our variants, variant 1 performs better overall than variant 2 over time.

Examining the random walk scenario as shown in fig. 4, we can see the impact a movement model has on network performance reported by our metrics. This is induced by the random nature of node movement in this scenario: While nodes traveled along well-defined paths in the map-based movement scenario they only had access to certain parts of the simulation area. In contrast to this, nodes now using the random walk movement can reach every point of the simulation area. Encounters happen less often, resulting in a lower overall congestion and our probabilistic decision for packet transmission is occasionally too restrictive for the detected node environment. This is especially clear in the scenario with only 10 nodes, as the average hop count is 1 for all examined algorithms as only direct deliveries take place.

What is also interesting to see is that in figs. 4a, 4b and 4d the behavior of our variants and Epidemic is about the same. Note the x-axis of fig. 4a showing only values between 0 and 20. We conclude that our Congestion Control algorithms identify node surroundings as not congested and behave almost like regular Epidemic routing. The similar overhead ratio suggests that as Epidemic, our algorithms are able to take advantage of available network resources, too.

The average latency of our algorithms shown in fig. 4c differs more from Epidemic than the results of the other metrics displayed. Because occasionally packets are still prevented from being transmitted when using our algorithms, average latency tends to become smaller compared to Epidemic as other packets are preferred for sending. This leads to locally generated messages being sent out earlier than with Epidemic, resulting in the lower average latency we can observe. The up-and-down pattern occurs due to node movement for the random walk.

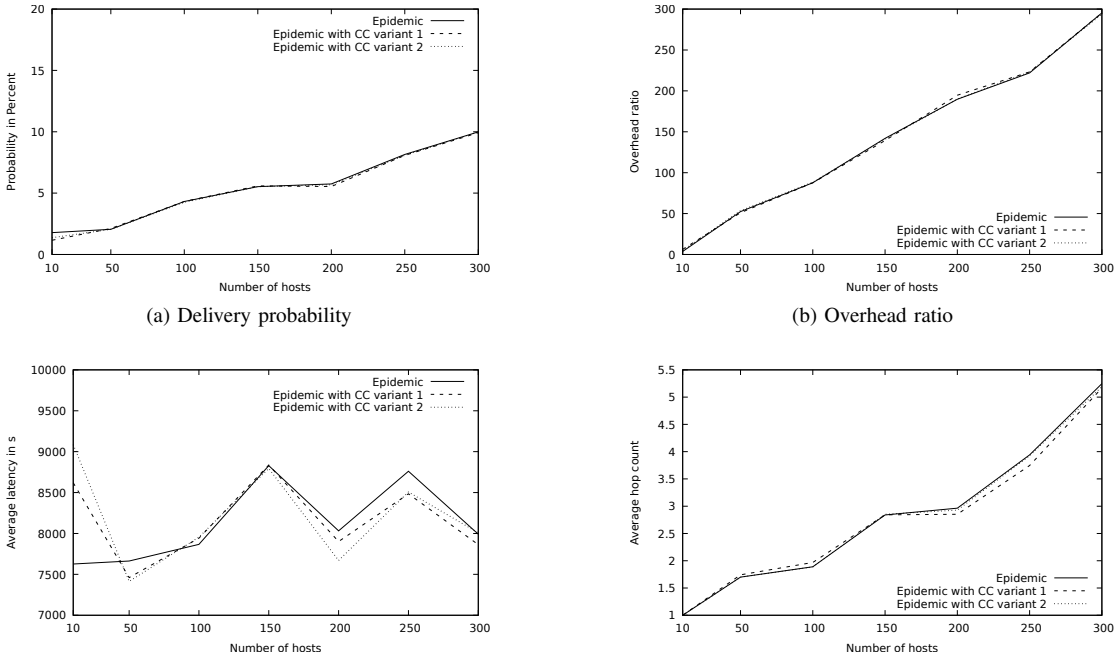
Concluding from the metric outcomes, our Congestion Control strategies perform at least about the same as Epidemic with significant improvements in scenarios with higher node density and thus higher congestion.

E. Discussion

We tested plain Epidemic routing and our Congestion Control strategies using two variants of a fitness function for transmission decisions. Employing different node counts leading to network congestion, we are able to show how our algorithms improve network performance with increasing congestion. Additionally, we conducted simulations using the same simulation scenario settings with a different movement model. This was conducted to exclude influence from node movement on simulation outcomes as well as to investigate our strategies for different situations of network congestion.We improved network performance and usage for congested environments, as indicated by the defined metrics, by employing our algorithms.

Our examinations also exposed the following two findings:

a) A certain node density (around 3.27 per km^2 in our simulations) inducing congestion enables our strategies to improve other network performance metrics in addition to reducing overhead as compared to Epidemic.



(a) Delivery probability

(b) Overhead ratio

(c) Average latency

(d) Average hopcount

Figure 4. Metrics output on Epidemic routing with and without congestion control strategies for the random walk movement scenario

b) Both our variants of our strategies can improve the outcome of examined metrics. With increasing node count, the outcomes diverge and our first variant provides superior metric measurements.

This is important when choosing a fitting Congestion Control strategy for a routing algorithm in a specific use case. While our algorithms show a general improvement in addition to a reduced overhead, they were designed to restrict a nodes' transmissions to provide increased message throughput for other nodes. When network utilization is already low, our algorithm cannot enable higher network utilization than Epidemic as there are no more messages to forward or exchange, that could lead to an increased network performance.

V. Conclusion

With this paper we provide two variants of Congestion Control strategies, both utilizing and enabling Epidemic routing to reach higher network performance. Using network performance metrics, we conclude that our initial goal of lowering network overhead and thus reducing congestion, as well as improving general network performance by also increasing delivery probability, has been met. Not only do our strategies provide better network performance in situations with higher congestion, they also prove to provide similar performance as Epidemic in scenarios where node movement results in generally fewer node encounters and message forwarding.

References

[1] C. Scheideler, *Universal routing strategies for interconnection networks*. Springer, 2006, vol. 1390.

[2] L. Pelusi, A. Passarella, and M. Conti, "Opportunistic networking: data forwarding in disconnected mobile ad hoc networks," *IEEE communications Magazine*, vol. 44, no. 11, 2006.

[3] A. Vahdat, D. Becker *et al.*, "Epidemic routing for partially connected ad hoc networks," *Technical Report CS-200006, Duke University*, 2000.

[4] B. Poonguzharselvi and V. Vetriselvi, "Survey on routing algorithms in opportunistic networks," in *Computer Communication and Informatics (ICCCI), 2013 International Conference on*. IEEE, 2013, pp. 1–5.

[5] H. Al-Bahadili, *Simulation in Computer Network Design and Modeling: Use and Analysis*, 1st ed. Hershey, PA, USA: IGI Global, 2012.

[6] I. Parris and T. Henderson, "The impact of location privacy on opportunistic networks," in *World of Wireless, Mobile and Multimedia Networks (WoWMoM), 2011 IEEE International Symposium on a*. IEEE, 2011, pp. 1–6.

[7] A. Dvir and A. Vasilakos, "Backpressure-based Routing Protocol for DTNs," in *SIGCOMM Computer Communication Review - CCR*, vol. 40. ACM, 2010, pp. 405–406.

[8] M. Radenkovic and A. Grundy, "Efficient and adaptive congestion control for heterogeneous delay-tolerant networks," *Ad Hoc Networks*, vol. 10, no. 7, pp. 1322–1345, 2012.

[9] A. Ippisch, S. Sati, and K. Graffi, "Optimal Replication Based on Optimal Path Hops for Opportunistic Networks," in *Proceedings of the IEEE International Conference on Advanced Information Networking and Applications*, 2018, pp. 1–8.

[10] S. Sati, A. Ippisch, and K. Graffi, "Replication Probability-based Routing Scheme for Opportunistic Networks," in *Proceedings of the International Conference on Networked Systems*, 2017, pp. 1–8.

[11] L. Yun, C. Xinjian, L. Qilie, and Y. Xiaohu, "A novel congestion control strategy in delay tolerant networks," in *Future Networks, 2010. ICFN'10. Second International Conference on*. IEEE, 2010, pp. 233–237.

[12] S.-C. Lo and Y.-S. Luo, "A neighbor-aware congestion control mechanism for delay-tolerant networks," in *Advances in Intelligent Systems and Applications-Volume 1*. Springer, 2013, pp. 591–600.

[13] S. Sati, C. Probst, and K. Graffi, "Analysis of buffer management policies for opportunistic networks," in *2016 25th International Conference on Computer Communication and Networks (ICCCN)*, 2016, pp. 1–8.

[14] A. Keränen, J. Ott, and T. Kärkkäinen, "The ONE Simulator for DTN Protocol Evaluation," in *SIMUTools '09: Proceedings of the 2nd International Conference on Simulation Tools and Techniques*. New York, NY, USA: ICST, 2009.