Congestion Control by Deviation Routing in Electromagnetic Nanonetworks

Thierry Arrabal
thierry.arrabal@univ-fcomte.fr
FEMTO-ST Institute, Univ. Bourgogne Franche-Comté, CNRS
Montbéliard, France

Florian Büther
buether@itm.uni-luebeck.de
University of Lübeck
Institute of Telematics
Lübeck, Germany

Dominique Dhoutaut
dominique.dhoutaut@univ-fcomte.fr
FEMTO-ST Institute, Univ. Bourgogne Franche-Comté, CNRS
Montbéliard, France

Eugen Dedu
eugen.dedu@univ-fcomte.fr
FEMTO-ST Institute, Univ. Bourgogne Franche-Comté, CNRS
Montbéliard, France

ABSTRACT

Pulse-based wireless nanonetworks differ in many ways from traditional wireless networks. This paper investigates congestion in multi hop nanonetworks, which do not behave as usual due to the specifics of the channel and the physical layer. Most protocols and network models assume that each node can listen to all of the traffic sent on their channel. In wireless nanonetworks, the capacity (in the order of Tb/s) of the shared channel is well beyond the processing capability of an individual node. Consequently, congestion arises from the limited buffers of individual nodes on the path instead of limited channel bandwidth.

After defining congestion in this context, we propose a solution suitable to large, wireless nanonetworks. Instead of decreasing the sending rate to reduce overall traffic and congestion, we use the SLR routing protocol to find less saturated routes. Our evaluation demonstrates the effectiveness of this solution and shows that the throughput can be preserved with moderate activity overhead and latency cost.

CCS CONCEPTS

• **Networks** → **Network protocol design**; **Routing protocols**; *Control path algorithms*; *Mobile ad hoc networks*; • **Hardware** → Nanoelectromechanical systems.

KEYWORDS

Congestion; Deviation; Routing; Nanonetworks

ACM Reference Format:
Thierry Arrabal, Florian Büther, Dominique Dhoutaut, and Eugen Dedu. 2019. Congestion Control by Deviation Routing in Electromagnetic Nanonetworks. In *The Sixth Annual ACM International Conference on Nanoscale Computing and Communication (NANOCOM '19), September 25–27, 2019, Dublin, Ireland.* ACM, New York, NY, USA, 6 pages. https://doi.org/10.1145/3345312.3345465

NANOCOM '19, September 25–27, 2019, Dublin, Ireland

ACM ISBN 978-1-4503-6897-1/19/09…$15.00
https://doi.org/10.1145/3345312.3345465

1 INTRODUCTION

Nanonetworks are made up of tiny, autonomous robots with a size measured in nanometres. These nanoscale robots, or *nanobots*, usually possess sensors and actuators, a processor and memory. They can move and communicate with each other [8]. A single robot is minuscule, no larger than a few micrometres. It will thus be limited in its computing capabilities, and needs to collaborate with other nanobots to fulfil its tasks [1]. This gives rise to *nanonetworks*.

Physical communication in nanonetworks employs electromagnetic waves in the terahertz band [13]. Nodes use Time Spread On-Off Keying (TS-OOK) [11] for medium access. Messages comprise a series of one pulse per bit spread out over time. TS-OOK can interleave messages: As pulses spread out, the channel is available to other messages in between bits.

Still, nanobots possess limited energy supplies, and thus can only communicate over short distances. Nanonetworks thus need to organise into multi hop networks, especially if they need to cover large areas: For example, a medical nanonetwork might need to cover a whole human body [18]. An environmental monitoring network to detect airborne contaminants might need to observe an even larger volume. Smart materials, which to a large degree are made up of nanobots [6], additionally exhibit a high network density.

With possibly thousands of neighbours, a nanobot cannot easily maintain an up-to-date list of its neighbourhood, and even less maintain IP-like routes to distant nodes. Alternative addressing schemes must use other approaches, for example a form of geocasting or spatial address [17]. A routing algorithm with spatial addressing can forward messages "in the direction" of a destination. It can thereby avoid the broadcast storm [19] that may easily arise with an electromagnetic channel. At the same time, spatial addresses allow simple forwarding schemes suitable for resource constrained nanobots.

Stateless Linear Routing (SLR) [20] uses the hop count as a measure of spatial distance. Several anchor nodes span a relative coordinate system, where each node stores the number of hops to all anchor nodes as its address. To forward a message, SLR computes if a node's address is on a line from source to destination coordinate. Nodes thus do not explicitly construct an end-to-end route, but compute just a local forwarding decision.

The narrow forwarding path is susceptible to congestion: If all nodes in an area along the path are busy with other communication, they will ignore new incoming packets. The forwarding chain collapses, and the message is lost.

Classic approaches to congestion control employ end-to-end schemes, which require expensive acknowledgement messages and conflict with SLR's local forwarding concept. Newer approaches, for example in wireless sensor networks, require extensive knowledge on each node difficult in nanonetworks. Moreover, as congestion in nanonetworks behaves differently, existing approaches on lower levels cannot apply here.

While the literature has previously addressed MAC and routing layers in nanonetworks, this paper is, to the best of our knowledge, the first to address congestion control and transport layer features in these networks. We first define congestion as it occurs in nanonetworks. We present a novel solution for nodes to locally detect this congestion. We then handle it by adopting a new SLR path that routes around the congested area. Finally, we evaluate this solution through detailed simulations.

To detect congestion and deviate packets, our algorithm leverages information inherent to TS-OOK as well as information already created by SLR. Our evaluation shows that the algorithm is able to successfully deviate a path to avoid a congested area, and can successfully deliver a message where original SLR fails. On a simple but meaningful scenario, we show that deviating packets preserves flow throughput. The deviated route is longer and so more nodes have to forward the packet (about 55 %). Nevertheless, the increased energy cost is alleviated as it is distributed over more nodes.

It is worth noting that most congestion control schemes react to congestion by reducing traffic at its source. Distinctively, deviating traffic to less congested areas allows to preserve its bandwidth. Deviating traffic is however not applied in most wireless networks, as they are mostly 2-dimensional. Except for very simple or specific scenarios, flows tend to cross each other, which eliminates most opportunities for traffic deviation. Contrarily, many types of wireless nanonetworks will be 3-dimensional, making deviation techniques such as the one presented in this paper very promising.

2 RELATED WORK

Congestion is a state of a network where the exchanges are so numerous that they block each other and prevent the system from fulfilling its objective. Well-known examples of congestable systems are car traffic (traffic jams) and air traffic (overcrowded airports). In these cases, common solutions are to either deviate some traffic or to delay it.

Similarly, congestion in networks describes a state where so many packets are exchanged that they are lost in the network [12]. If a router receives more packets than it can send on an outgoing channel, it queues them in a buffer. However, if even this buffer is saturated, the router starts to drop further packets. Alternatively, routers may drop random packets *before* congestion, such as in RED (random early detection) policy [7] and its variants. Additionally, in wireless networks, senders need to contend for channel access, which may cause collisions and packet loses.

The mechanisms to avoid and recover from such losses are grouped under the terms *congestion control*. In Internet macroscale networks, the congestion control is handled *end-to-end*, that is, only by the endpoints. When a packet is lost by the widely-used TCP protocol, the sender assumes that a link on the path is congested and slows down its traffic.

A different approach has been taken with the proposal of the ECN extension of TCP/IP [15], where routers are involved in congestion control. Still, their involvement is very limited, as their only task is to mark packets in a pre-congestion phase, and eventually the same action is taken: the sender decreases its throughput.

Also, in wireless networks, the bit-error rate is usually high and loses are incorrectly interpreted as congestion.

It is worthwhile to note that congestion avoidance and recovering increases *total transmission time*. This is because the only response the senders have for congestion is to reduce their throughput. Even worse, congestion control is delayed by a full round trip, as the receiver has to inform the sender of the congestion.

Recently, the growing interest in different types of network, such as mobile ad hoc networks (MANET) and wireless sensor networks (WSN), lead to proposal of new techniques of congestion control.

The parameters used to detect and avoid congestion in MANETs are specific to the link layer. For example, a routing-based load balancing mechanism in MANET is proposed in [2]. Congestion is detected using three parameters: Available bandwidth based on channel saturation, load estimation using the contention window of Wi-Fi packet transmission, and residual energy of node. Upon congestion detection, packet deviation is done with route discovery. The detecting router acts on congestion, similar to our approach, but none of the methods used apply to nanonetworks: The available bandwidth is not a concern in nanonetworks, but the buffer memory is; there is no Wi-Fi-like retransmission, transmission energy cost computes differently, as sending a 0 bit does not consume energy; and there is no route discovery at all in SLR nanonetworks.

Load balancing in LARA [16] uses the degree of contention at the MAC layer. Among other constraints, it requires that each node maintains a record of the latest traffic queue estimations at each of its neighbours. Nanonodes have too little capacity to store the required information, especially in dense neighbourhoods.

Multipath routing in WSN [3] is similarly not possible in nanonetworks. Nodes either need to know their neighbourhood, or the congestion detection techniques used do not map to nanonetworks.

3 CONGESTION IN NANONETWORKS

Properties of electromagnetic nanonetworks differ from macroscale wireless networks and affect protocols. Terahertz frequencies allow for higher bandwidth and throughput, up to several Tb/s [13]. Pulse-based sending, as in TS-OOK, can interleave bits of different messages, and thus dissolves the sequential nature of channel access and message reception. Finally, the limited resources of nanobots, especially limited memory and processing power, decrease the ability to process incoming packets. As explained below, congestion in nanonetworks will not arise from a saturated channel, but rather from insufficient capacity of single nodes on the path (routers) to process all the incoming concurrent packets from various flows.

Pulse-based sending allows for multiple packets to be interleaved, and nodes can concurrently receive them. The amount of concurrent transmissions only depends on the time interval between two

consecutive bits of a packet. The channel itself poses no limit. This type of *packet interleaving* is very different from the classical TDMA scheme, where *flows* are interleaved. In TDMA, only one packet is over the air at any given time. Moreover a node willing to send a lot of data could theoretically use all slots and use the full bandwidth of the channel. It could also choose to listen to all the transmissions from its neighbourhood.

In pulse-based nanonetworks, there are hardware limitations to the number of interleaved packets that a node can concurrently track and process, *de facto* limiting the possible throughput of the node. The limitations could be:

- Insufficient hardware, preventing the reception of many concurrent packets.
- Limited memory or limited processing power preventing nodes to store or process packets as fast as they arrive. Even if most incoming packets are not addressed to a node, it has at the very least to process their headers to decide.
- Limited energy restricts prolonged operation of a node, especially considering the small size of node batteries and their expected slow recharge rate.

Moreover, a single transmitting nanonode is not expected to be able to consume all the available bandwidth by itself (due to processing power and energy considerations).

As a consequence, there is a disparity between the raw channel capacity and the effective channel activity a node can handle. As the number of concurrent transmissions increases, it will get closer to and eventually exceed the capability of the nodes, while still being well below the channel capacity.

The nodes are aware of their limitations and can monitor the number of concurrent packets being received, translating it into the node's actual congestion level. Moreover, due to the broadcasting nature of the channel, detecting this type of congestion at a node means that neighboring nodes are most probably experiencing the same conditions. This is fundamental in our work, as it means that the area near a node is congested as a whole, and that it may be for the best to avoid routing packets through it.

4 ROUTING WITH SLR BACKOFF FLOODING

We use a modified version of the SLR routing protocol extended by backoff flooding. SLR computes a packet route, and backoff flooding reduces the number of forwarders.

4.1 Stateless Linear-Path Routing

Stateless linear-path routing [20] is an addressing and routing protocol designed for nanonetworks. It is divided into two phases.

The first phase assigns addresses to all nodes. Special anchor nodes each initiate the broadcast of a beacon to the whole network. A 2D network requires two anchor nodes (placed at the edges of the network), whereas a 3D network requires at least 3 anchors. Beacon messages include a hop distance field initialised to 0, which increments with each retransmission. Nodes adopt the value from this field as their distance to the respective anchor. The distances to all anchors create a coordinates system that nodes can use to route packets. Note that this process does not assign a unique address to each node; instead, it creates *zones* where multiple nodes share the same coordinates.

In the following routing phase, packets contain the SLR address of the sender and receiver. When a node receives a message, it uses a simple formula [20] to check if it is itself located on the path from the source to the destination, and only in this case forward the message. The check uses only basic integer computations, appropriate to the low computation capabilities of nanonodes.

SLR can create paths of different width. It specifies a parameter m, which configures the width of the SLR path as a number of zones. By default, m equals 1, which makes nodes forward the packet only if they are in a zone directly on the path. Increasing m widens the path, and spreads the forwarding area over more zones.

4.2 Backoff flooding

In multi hop diffusion, depending on the network density and communication scheme, the well known broadcast storm problem may arise. If nodes immediately forward all packets they receive, many copies may be sent in a very short amount of time. Firstly, this will be energetically inefficient. Secondly, this will lead to packet losses through collisions, but even more through saturation of the reception capabilities of the nodes. As seen in Section 3, nodes have to store and decode all packets in order to check whether all these packets are copies of the same packet.

Backoff flooding [5] is a multi hop diffusion protocol. It is designed to drastically reduce the number of transmitters while maintaining full coverage. When a node receives a packet for the first time, it does not forward it immediately, but instead initiates a random duration timeout in a backoff window. If the node receives enough copies of the packet from other nodes before the timeout, the node discards it, otherwise it will forward the packet. In both cases, the node must memorise the ID of the packet so that it does not consider any subsequent copies.

The backoff window has to take the local network density into account. The size of the backoff window is discussed in [5]. Here, we also use DEDeN [4] to easily obtain the local estimation of the network density.

Multi hop broadcasting methods tend to randomly select forwarders, so non-optimal ones may be chosen. In the worst case, a forwarder may not forward the packet to new nodes, effectively halting its propagation. To avoid this, backoff flooding uses a packet counter as a *redundancy* parameter, which guarantees that a message is forwarded at least a given number of times. This aspect has been further studied in [5]. This paper uses a redundancy of 1.

4.3 Merging the two solutions

Section 3 emphasises the preponderance of node limitations over raw channel capacity in the appearance of congestion. Consequently, monitoring the degree of saturation of a node is of key importance in our work.

With original SLR, *all* nodes from zones on the path forward a packet. In dense zones, this leads to congestion for even one message initially sent. To avoid this situation, we combine SLR with backoff flooding. As such, only very few nodes in each SLR zone forward the packet.

Using backoff flooding highly reduces the number of packets sent to transmit a message. This helps to reduce the congestion since it makes better use of network resources. Additionally, it has

a positive impact on energy consumption. Finally, it also prevents nodes from filling their reception capabilities with only 1 message.

The second modification to SLR concerns the zone size. Nodes might not be able to count all packets sent by a neighbouring zone, which leads to unnecessary retransmissions. Therefore, we modified the SLR addressing phase by making nodes use only a fraction of their sending power, and hence smaller communication range. This generates smaller zones, which facilitates packet counting in backoff flooding.

We refer to this new protocol as *modified SLR*.

5 CONGESTION DETECTION

The main type of congestion in nanonetworks, as given in Section 3, is congestion of reception buffers of intermediate nodes (routers). Luckily, the number of buffers currently in use is readily available to a nanobot. A nanobot can add a single bit flag to each reception buffer, which it sets when starting to receive a message and unsets when it finishes reception. It's worth noting that contrary to end-to-end congestion detection, this method does not misinterpret loses caused by bad channel conditions as congestion.

A nanobot n can estimate its local congestion at a time t as follows:

$$c_n(t) = r_n(t)/r_{\max}$$

Here, $r_n(t)$ is the number of reception buffers of node n in use at time t, and $r_{\max}$ is the overall number of reception buffers of a node.

The resulting congestion quota c_n denotes a *level* of congestion: The node can detect a partial congestion even before a full congestion blots out all remaining traffic. The congestion quota may as well be interpreted as a spatial property: Nodes close to a busy area of the network will exhibit a high congestion level, which declines as a node's distance to the area increases.

Several thresholds of the congestion quota can classify the state of the channel around the node. If the congestion level rises above the congestion threshold c_U, the network may soon completely congest, and preemptive measures might need to be taken. A congestion level c_L or less conversely indicates an area of little risk of congestion. In this case, a node can forward messages as usual, or even switch to an optimised forwarding scheme.

A very low number of reception buffers $r_{\max}$ may preclude congestion detection, if c_U grows larger than $(r_{\max} - 1)/r_{\max}$. In this case, a node will only be able to detect congestion when the network is already fully congested. However, even if it sends a message, other nodes likely cannot receive any more messages as well.

Section 7 will show that routing behaviour can improve even with very few reception buffers. A larger $r_{\max}$ additionally elevates the congestion estimate from a yes/no information to a gradient. Subsequent algorithms may then make use of the gradient, for example to introduce a longer resending delay, or deviate more strongly from the current position.

6 PACKET DEVIATION

When a nanobot on the path detects increased congestion, it tries to react by *deviating* its current packet. It still employs the SLR path as a basis, but tweaks the path width: If the path width increases part of the way, the message "spreads out" to occupy additional zones.

Figure 1 visualises the various cases. Each subfigure shows the SLR zones in the background as a wave-like pattern. These zones are the result of the beacon phase of SLR, where each node determines its distance by hop count to both anchors. On top of the zones, nodes that forward the message are shown in dark blue, and nodes that have reached $r_{\max}$ are shown in bright yellow.

Figure 1a shows an example of routing without network congestion. The SLR protocol [20] routes packets following a "linear" path from source SLR coordinate to destination SLR coordinate. m is 1, so the path is one zone wide.

Nodes decide whether to forward a packet or not using the `isOnPath()` function. A node checks on each successful reception if it is on the path and should forward the message:

```
bool isOnPath(node n, address src, address dst, int m):
  return (n is on SLR path of width m from src to dst)
```

The message in Figure 1a originates from the bottom, and propagates through the network towards the destination zone at the top of the picture. In several zones, very few nodes resend the message. This is caused by backoff flooding, described in Section 4.2: After nodes receive more copies of the message, they reach the specified redundancy and do not forward the message.

In Figures 1b and 1c, the yellow area is congested, and nodes in that area cannot receive further messages. Figure 1b shows nodes forwarding with modified SLR, which does not consider congestion. Consequently, it cannot forward through the congested area, and the message is lost.

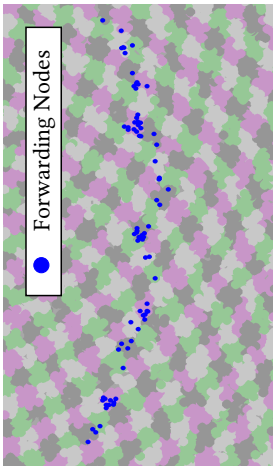
We propose to use the congestion level c_n as an indicator for congestion. As soon as $c_n(t) > c_U$, nodes deviate the message route away from the current path. To do so, the packet header stores an additional deviation value, which corresponds to the m value of SLR. This value is initially set to 1 as long as nodes do not detect congestion. On congestion, a node increments this value and forwards the message. Further nodes then override the path width m with the deviation value, causing the message to spread out.

However, the deviation algorithm should not occupy *more* zones to forward packets, as that would further increase congestion. Instead, we want *different* zones to forward packets. We thus adopt a modified `isOnPath()` function as follows:

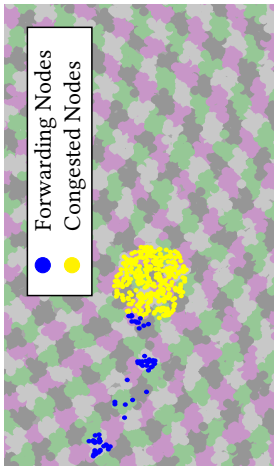
```
bool isOnPathDeviation(n, src, dst, m):
  return isOnPath(n,src,dst,m) AND
    NOT isOnPath(n,src,dst,m-1)
```

This function forwards packets to the edges of a path of width m. As long as nodes detect congestion, they increase m, thereby moving the packet further and further from the congested area. Congestion detection along with SLR path deviation comprises our proposed *deviation SLR* protocol. In Figure 1c, deviating SLR successfully detects the congestion and starts to deviate the message.

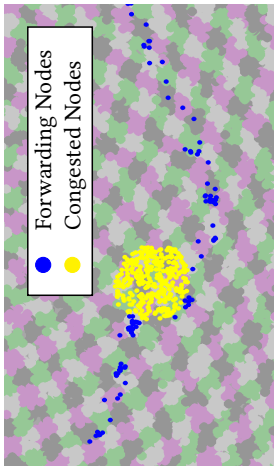
Eventually, the packet reaches nodes with $c_n(t) < c_L$, where the local congestion is less than the lower congestion limit. These nodes now reduce the deviation value of the packet until it again reaches 1 (or the packet encounters further congestion). Additionally, the nodes set their local zone as a new intermediate source `src'`, which acts like `src` for further routing. This causes the packet to route on a new straight line towards `dst`. The effect can be observed in Figure 1c, where slightly above the congestion the deviation ceases and the message continues in a straight line towards its destination.



(a) No congestion: Messages forward on an SLR path.



(b) Congestion with SLR: The message cannot pass the congestion.



(c) Congestion with deviation: Routing deviates as it encounters the congestion.

Figure 1: Comparison of routing schemes in the presence of congestion. Messages propagate from the bottom upwards.

Given that our proposal uses simple integer divisions and comparisons, it is appropriate to devices with limited capabilities, such as nanonodes.

7 EVALUATION

We evaluate our proposed solution by simulation. The algorithm depends on the interaction of large numbers of nodes, as they are typical in nanonetworks: The interaction between interleaving messages and the limit on concurrent receptions may result in emergent behaviour that is not obvious from the algorithm itself. Plainly, we need a high number of nodes simply to have network parts to deviate into.

Several choices for nanonetwork simulators exist: COMSOL Multiphysics[1], a physics simulator, provides a precise simulation, but can simulate only a few nodes. Nano-Sim [14] and TeraSim [10] are two extensions to the well-known network simulator ns-3[2]. They are able to handle up to around 1000 nodes, which we still consider insufficient for deviation tests in high density nanonetworks. We previously developed BitSimulator [9], a simulator specifically tailored to dense nanonetworks with more than 100 000 nodes. BitSimulator already implements the SLR protocol and provides configurable buffer sizes for nodes. Our evaluation thus uses BitSimulator. We provide technical details and information about reproducibility of our results on a separate website[3].

We evaluated the congestion detection and deviation algorithm in a simulation of a 2D, dense nanonetwork. The 2D simulation only allows two directions to deviate towards, which exhaust quickly in complex scenarios. Real nanonetworks are most likely 3D, where deviation has many more possible routes available.

Table 1 lists the parameters of our simulation. Note that packet size (100 bits) has no influence on the deviation. $r_{\max}$ is 5 for all nodes, so nodes can receive no more than 5 messages concurrently. c_L and c_U are both 0.5. A node thus increases the deviation value starting with the third message it receives concurrently.

Table 1: Simulation parameters.

Size of simulated area	6 mm ∗ 6 mm
Number of nodes	20 000
Communication distance	350 µm
Average number of neighbours	203
β (TS-OOK time spread ratio)	1000
Packet size	100 bit
SLR redundancy	1
$r_{\max}$	5
c_L/ c_U	0.5 / 0.5

The evaluated scenario sends a single message across the network, using either modified SLR, or modified SLR augmented with congestion detection and deviation. In the first case, the network is completely free of other traffic. In the second case, an area directly in the SLR path of the first case is congested. We evaluate each case with 10 runs and average the results over all runs. Each run uses a different seed for the random waiting times of SLR backoff and for density estimation, all other parameters are identical.

Figure 1 shows the simulated scenarios. The message propagates upwards, from the initial source at the bottom towards a receiving SLR zone at the top end of the shown network.

First and foremost, the deviation algorithm works as expected in a network with congestion, i.e. correctly detects the congested zone, deviates the packets around it, and routes the message successfully to the destination zone in each run, similarly to the same figure 1.

We then compare modified SLR and deviating SLR in the number of messages sent by all nodes as well as the time elapsed between initial send and reception at the destination. Table 2 shows the

[1] https://www.comsol.com/
[2] https://www.nsnam.org/
[3] BitSimulator Web page: http://eugen.dedu.free.fr/bitsimulator

Table 2: Evaluation results for both routing algorithms.

	Avg. packets sent	Avg. elapsed time
Without congestion:		
Modified SLR	96.4	2.72 µs
Deviating SLR	87.3	2.63 µs
With congestion:		
Modified SLR	–	–
Deviating SLR	135.0	4.10 µs

results of the evaluation. Modified SLR and deviating SLR show very similar results for an uncongested network, which only differ due to the random waiting times of SLR backoff.

In a network with congestion, it required 55 % more messages and 56 % more time relative to forwarding without congestion. Modified SLR was not able to route the message in a congested network, thus no results are available for this case.

Figure 1 shows that deviation happens as part of the routing algorithm: Neither sender nor receiver can detect that a message deviated during its transmission. This implies that deviating SLR cannot reduce traffic as for example TCP rate limiting does. As explained in Section 2, this approach trades network-wide energy cost for low transmission delays even in the case of congestion. It further performs implicit load balancing, shifting forwarding towards less utilised areas of the network.

8 CONCLUSION

This paper presents a novel algorithm to detect congestion within a nanonetwork and deviate a routing path away from the congestion. It uses the concurrent nature of TS-OOK to measure congestion as the relation of current to maximum possible parallel receptions. If congested, it exploits the path width m of SLR to deviate the route to reach less utilised areas of the network.

Our evaluation shows that this approach successfully delivers messages that otherwise are lost due to congestion. Deviation happens as part of regular forwarding, so messages only deviate if needed, and requires no additional messages for bookkeeping.

We validated our approach using a detailed simulation. In a 2D validation scenario, deviating around a congested area increased the total number of messages sent by 55 %, and they took 56 % longer to deliver. Both figures are lower than the usual cost of collisions and retransmissions. If no congestion is present, our algorithm does not change the SLR path, and forwarding is as efficient as modified SLR. We expect that our proposal is more successful in 3D networks, where many opportunities of non-crossing routes exist. In this case, a large number of concurrent communications would give more insights into the effectiveness of our proposal.

Detecting a congestion and deviating around it are algorithmically separate actions: They may be combined with other algorithms, or be applied to different challenges, for example, to deviate around a network hole in the linear path. Furthermore, deviating SLR may behave differently if the network is already fully congested. Further research into other applications and scenarios is warranted.

ACKNOWLEDGMENTS

Thierry Arrabal has a grant from Pays de Montbéliard Agglomération. We thank Tucker Wilson for a preliminary work on this topic.

REFERENCES

[1] Ian F Akyildiz and Josep Miquel Jornet. 2010. The internet of nano-things. *IEEE Wireless Communications* 17, 6 (2010), 1–6.

[2] M. Ali, B. G. Stewart, A. Shahrabi, and A. Vallavaraj. 2012. Congestion adaptive multipath routing for load balancing in mobile ad hoc networks. In *8th International Conference on Innovations in Information Technology (IIT)*. IEEE, Abu Dhabi, United Arab Emirates, 305–309.

[3] Aboli Arun Anasane and Rachana Anil Satao. 2016. A Survey on various Multipath Routing protocols in Wireless Sensor Networks. In *International Conference on Communication, Computing and Virtualization (ICCCV)*. Elsevier, Mumbai, India, 610–615.

[4] Thierry Arrabal, Dominique Dhoutaut, and Eugen Dedu. 2018. Efficient Density Estimation Algorithm for Ultra Dense Wireless Networks. In *27th International Conference on Computer Communications and Networks (ICCCN)*. IEEE, Hangzhou, China, 1–9.

[5] Thierry Arrabal, Dominique Dhoutaut, and Eugen Dedu. 2018. Efficient multihop broadcasting in dense nanonetworks. In *17th IEEE International Symposium on Network Computing and Applications (NCA)*. IEEE, Cambridge, MA, USA, 385–393.

[6] Julien Bourgeois, Benoit Piranda, Andre Naz, Nicolas Boillot, Hakim Mabed, Dominique Dhoutaut, Thadeu Tucci, and Hicham Lakhlef. 2016. Programmable matter as a cyber-physical conjugation. In *Systems, Man, and Cybernetics (SMC), 2016 IEEE International Conference on*. IEEE, Budapest, Hungary, 002942–002947.

[7] Bob Braden, David Clark, Jon Crowcroft, et al. 1998. Recommandations on Queue Management and Congestion Avoidance in the Internet. IETF standard. RFC 2309.

[8] Florian Büther, Florian-Lennert Lau, Marc Stelzner, and Sebastian Ebers. 2017. A Formal Definition for Nanorobots and Nanonetworks. In *Internet of Things, Smart Spaces, and Next Generation Networks and Systems: 17th International Conference, NEW2AN 2017, 10th Conference, ruSMART 2017, Third Workshop NsCC 2017, St. Petersburg, Russia, August 28–30, 2017, Proceedings*, Olga Galinina, Sergey Andreev, Sergey Balandin, and Yevgeni Koucheryavy (Eds.). Springer International Publishing, Cham, 214–226. https://doi.org/10.1007/978-3-319-67380-6_20

[9] Dominique Dhoutaut, Thierry Arrabal, and Eugen Dedu. 2018. BitSimulator, an electromagnetic nanonetworks simulator. In *5th ACM/IEEE International Conference on Nanoscale Computing and Communication (NanoCom)*. ACM/IEEE, Reykjavik, Iceland, 1–6.

[10] Zahed Hossain, Qing Xia, and Josep Miquel Jornet. 2018. TeraSim: An *ns-3* extension to simulate Terahertz-band communication networks. *Nano Communication Networks* 17 (Sept. 2018), 36–44.

[11] Josep Miquel Jornet and Ian F Akyildiz. 2011. Information capacity of pulse-based wireless nanosensor networks. In *8th Annual IEEE Communications Society Conference on Sensor, Mesh and Ad Hoc Communications and Networks (SECON)*. IEEE, Salt Lake City, UT, USA, 80–88.

[12] James F. Kurose and Keith W. Ross. 2003. *Computer Networking: A Top-Down Approach Featuring the Internet*. Pearson Education, Inc.

[13] Josep Miquel Jornet Montana. 2013. *Fundamentals of Electromagnetic Nanonetworks in the Terahertz Band*. Ph.D. Dissertation. Georgia Institute of Technology.

[14] Giuseppe Piro, Luigi Alfredo Grieco, Gennaro Boggia, and Pietro Camarda. 2013. Nano-Sim: simulating electromagnetic-based nanonetworks in the network simulator 3. In *6th International ICST Conference on Simulation Tools and Techniques (SimuTools)*. ACM, Cannes, France, 203–210.

[15] K. Ramakrishnan, Sally Floyd, and David Black. 2001. The Addition of Explicit Congestion Notification (ECN) to IP. RFC 3168.

[16] Vikrant Saigal, Ajit K. Nayak, Sateesh K. Pradhan, and R. Mall. 2004. Load balanced routing in mobile ad hoc networks. *Computer Communications* 27, 3 (Feb. 2004), 295–305.

[17] Marc Stelzner, Falko Dressler, and Stefan Fischer. 2017. Function Centric Nano-Networking: Addressing nano machines in a medical application scenario. *Nano Communication Networks* 14 (12 2017), 29–39. https://doi.org/10.1016/j.nancom.2017.09.001

[18] Marc Stelzner, Florian-Lennert Lau, Katja Freundt, Florian Büther, Mai Linh Nguyen, Cordula Stamme, and Sebastian Ebers. 2016. Precise Detection and Treatment of Human Diseases Based on Nano Networking. In *Proceedings of the 11th EAI International Conference on Body Area Networks (BodyNets '16)*. ICST, Brussels, Belgium, Belgium, 58–64.

[19] Yu-Chee Tseng, Sze-Yao Ni, Yuh-Shyan Chen, and Jang-Ping Sheu. 2002. The Broadcast Storm Problem in a Mobile Ad Hoc Network. *Wirel. Netw.* 8, 2/3 (March 2002), 153–167. https://doi.org/10.1023/A:1013763825347

[20] Ageliki Tsioliaridou, Christos Liaskos, Eugen Dedu, and Sotiris Ioannidis. 2017. Packet routing in 3D nanonetworks: A lightweight, linear-path scheme. *Nano Communication Networks* 12 (June 2017), 63–71.