Hop Count Routing: A Routing Algorithm for Resource Constrained, Identity-Free Medical Nanonetworks

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

Immo Traupe
traupe@itm.uni-luebeck.de
University of Lübeck
Institute of Telematics
Lübeck, Germany

Sebastian Ebers
ebers@itm.uni-luebeck.de
University of Lübeck
Institute of Telematics
Lübeck, Germany

ABSTRACT

Nanodevices, tiny robots operating within a human body, may help to detect and treat many kinds of diseases. As their individual abilities are limited by size, they need to work in concert. Communication provides the fundamental ability to enable this collaboration. In medicine, nanodevices act as a tool for a physician to report sensor data and receive action commands. Their communication thus flows to and from a gateway to the macro-world. Routing algorithms focus on enabling these data streams. We propose a new routing algorithm for medical nanonetworks based on a network topology constructed from the hop count distance to a single gateway. It exploits the distance as a direction indicator to deliver data towards or away from the gateway. The resource constrained nanodevices store no unique identity, but only require a single integer each. Simulation results show that a naive implementation produces an exponential number of messages. We mitigate this with a second approach by removing the hop count when retrieving sensor data, which requires only a linear number of messages. Our comparison finds the latter to be more efficient in terms of transmitted messages, while the first implementation is more suitable for routing several messages in parallel.

CCS CONCEPTS

• **Networks** → **Routing protocols**; **Mobile ad hoc networks**; *Wireless access points, base stations and infrastructure*; • **Applied computing** → *Life and medical sciences*;

KEYWORDS

Routing; Nanonetworks; Algorithm; Hop Count; Identity-Free

ACM Reference Format:
Florian Büther, Immo Traupe, and Sebastian Ebers. 2018. Hop Count Routing: A Routing Algorithm for Resource Constrained, Identity-Free Medical Nanonetworks. In *NANOCOM '18: NANOCOM '18: ACM The Fifth Annual International Conference on Nanoscale Computing and Communication, September 5–7, 2018, Reykjavik, Iceland.* ACM, New York, NY, USA, 6 pages. https://doi.org/10.1145/3233188.3233193

1 INTRODUCTION

Many of today's medical challenges involve processes at the molecular scale, for example combating bacteria causing pneumonia [15]. The idea of nanometre-sized computational robots, *nanodevices* [4], acting within the human body, poses a possible approach to many problems at the cellular or sub-cellular scale.

Nanodevices are restricted by size constraints arising from the deployment scenario. If used inside the human body, the size of the smallest capillaries provides a valid limit, which is about 4 μm [7]. Consequently, nanodevices are expected to work in cooperation with other nanodevices and more powerful devices, for example in a Body Area Network (BAN) or even in the internet, forming the Internet of Nano-Things (IoNT) [2]. These nanodevices collaborate in networks, and are thus called *nanonodes*.

Using the definition from [4], we assume nanodevices to be nanoscale, artificial, goal-oriented agents, possessing an energy supply and components for computation, actuation, sensing and communication. Due to the limited size, as well as the physical properties of nanoscale constructs, their computational capabilities are limited [8]. Additionally, nanonodes need to employ new technologies for communication suitable for nanonetworks. One major approach utilises electromagnetic waves in the terahertz spectrum [11], as emitted by graphene antennas. As terahertz waves can only be detected reliably at distances of about 2 mm within the human body [18], we assume that nanonetworks need to be mesh-shaped and require multi-hop forwarding solutions. Consequently, nanonetworks are composed of a large number of nanonodes to successfully deliver messages in a human body.

Nanonodes likely cannot fulfil the resource requirements of networking solutions for macroscale devices: Their memory may not be able to store complex routing tables, and they may not be able to compute complex forwarding decisions. Even worse, nanonodes may not even be able to store any kind of unique identifier, as the small amounts of available storage are required for functional applications and their algorithms.

This paper presents a routing algorithm based on directional message propagation, relying on distance measured by hop count. This routing scheme requires no individual identifiers or routing tables, thus eases the strain on the nanonodes' precious resources. We introduce, evaluate and discuss two variations of this routing algorithm and compare their message overhead and reliability.

2 MEDICAL NANONETWORKS

Medical nanonetworks are networks of nanonodes deployed *within* a human body. They consist of three tiers, as shown in Figure 1:

NANOCOM '18, September 5–7, 2018, Reykjavik, Iceland

ACM ISBN 978-1-4503-5711-1/18/09…$15.00
https://doi.org/10.1145/3233188.3233193

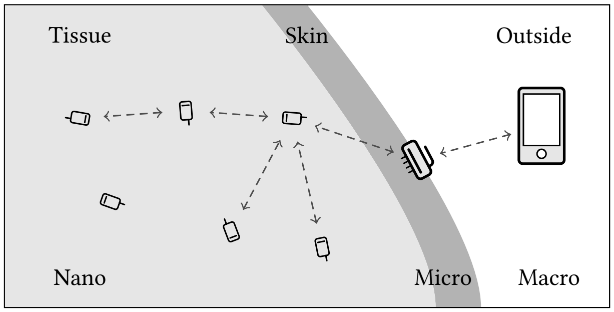

Figure 1: A 3-tiered medical nanonetwork architecture.

The nanonetwork itself, which connects to a microscale gateway attached to or implanted in the skin, which in turn connects to a regular computer in the BAN, for example a smart phone. Note that the micro-gateway is of special importance here, as it needs to mediate BAN communication, for example Bluetooth, and terahertz waves receivable by the nanonodes.

In a medical scenario, nanonodes will typically either sense a medical parameter, or manipulate their environment, for example by releasing a drug [15]. Sensor data needs to be passed to the macro-device, where it can be processed or inspected by a supervising physician, which in turn results in actuation commands that flow back into the nanonetwork. The medical nanonetwork thus exhibits two major flows of information: (1) Sensor information towards the gateway, and (2) Actuation commands from the gateway into the network.

Nanonodes in the body are nearly always in motion: If they operate within the natural blood stream, they are subject to the blood flow. Even if a device attaches to a bit of tissue, the person herself moves and thus changes the overall spatial relation. Consequently, it is impossible to maintain a fixed network structure, and nanonodes have to communicate only through spontaneous, peer-to-peer connections. We capture these aspects by interpreting medical nanonetworks as *mobile* and *ad hoc* networks [13].

2.1 Identity-Free Communication

Nanonodes possess only limited, if any, memory. Lau et al. [8] note that graph labelling, which in the context of networks is equal to assignment of unique ids, requires logarithmic space on each device. This corresponds to machines capable of complexity class L, which is a comparatively large class for nanodevices [8]. Consequently, unique identifiers for nanonodes are considered infeasible [8, 17]. Individual nanonodes are thus indistinguishable from some or all of their peers. Several consequences for routing arise:

- Unicast communication on shared channels is impossible, as messages cannot address a unique recipient.
- Similarly, nanonodes cannot reply specifically to the sender of a message.
- Nanonodes cannot collect identifying information about neighbours, like a list of directly reachable nodes.
- Nanonodes cannot detect their neighbourhood size, as a node cannot distinguish one talkative neighbour from several quieter ones.

Many well-known routing schemes conflict with these restrictions, as either the sender chooses recipients, or a receiver uses neighbourhood information to decide forwarding. An important remaining algorithm is broadcasting. The simplest example is unconditional broadcast, which however exposes the problem known as *broadcast storm* [16], where a single message claims large amounts of network resources.

2.2 Simplifying Assumptions

The described circumstances for medical nanonetworks pose strong limits on nanonode operations. To more easily derive a first algorithm for routing, we introduce additional assumptions about the setting:

- *Nanonodes have infinite energy.* Of course, nanonodes need to try to limit the amount of messages sent in order to conserve energy. We abstract this requirement, and only observe the overall message amount.
- *Nanonodes are not destroyed.* While nanonode degradation is inevitable at some point, we assume that the nanonodes operate perfectly for the time of usage. As nanonodes have no unique identity, loss of a specific node must be tolerable for the nanonetwork.
- *Nodes do not move.* Even though medical nanonetworks are inherently mobile, we assume the nanonodes to be static.
- *Communication is free of interference or loss.* Physical effects like molecular absorption may interfere with communication. Still, we assume that communication up to a specified distance always succeeds.

While these assumptions simplify the identified real-world conditions of medical nanonetworks, they still allow an evaluation to provide useful insights into algorithm performance and practical feasibility.

3 RELATED WORK

Tsioliaridou et al. [17] show a routing algorithm for a homogeneous nanonetwork with multiple (at least four) gateway-like anchors. Each node constructs its three-dimensional position from the hop count distance to these anchors, thereby allowing directed routing and localisation.

Liaskos and Tsioliaridou [9] propose a flood-based propagation scheme for a network of nanonodes allocated regularly on a grid. A single, predetermined node floods sensor data. Nodes use their spatial position to intelligently decide when to forward the message.

More generally, routing schemes based on hop counting can derive from generic routing schemes. For example, Link-State Routing, based on the Belmann-Ford-Algorithm [5], forms the basis for the well-known internet routing protocol RIP [10]. When each link is assigned a weight of 1, each router maintains a list of hop count distances to all targets.

4 HOP COUNT ROUTING

The shape of medical nanonetworks indicates that data flows only from or towards the microscale gateway. Nanonodes thus forward messages only in either of these directions, and only do so if they

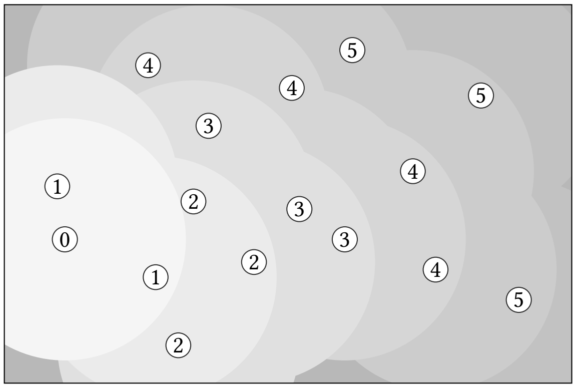

Figure 2: The gateway-oriented topology. Each node shows its assigned hop count. Lower hop counts are closer to the gateway and project a brighter communication area.

```
onReceiveMessage(Message m) {
  if (m.type == PropagationType &&
      this.hopCount > m.hopCount + 1) {
    this.hopCount = m.hopCount + 1;
    sendMessage(PropagationType,
      this.hopCount);
  }
}
```

Listing 1: Hop Count Propagation Algorithm

receive them from the respective other direction. For this, a nanonode must classify all neighbours as either closer to, at the same distance, or further apart from the gateway than itself.

The Hop Count Routing algorithm constructs a topology on top of the network where each nanonode stores the distance to the gateway in hops, as shown in Figure 2. This topology exhibits exactly the directional property noted above: Nanonodes with a smaller hop count lie in the direction of the gateway, denoted by hop count 0, whereas nanonodes with a larger hop count lie further away from it.

In a wireless network with a shared channel, all nodes in range receive a signal, similar to broadcast. The forwarding decision thus falls to the receiver, who inspects the message direction as well as the source hop count to decide.

4.1 Establishing the Hop Count

Initially, all nanonodes have an unknown distance to the gateway, their hop count k is ∞. To establish the hop count topology, the gateway starts a hop count propagation phase. It broadcasts a propagation message with its hop count 0 to its direct neighbours. Receiving nanonodes proceed as in Listing 1: If their own hop count is higher than the received hop count n_s, they update their hop count to $n_s + 1$ and continue the propagation by broadcasting their new hop count.

As all nanonodes initially have an infinite hop count, all nodes will update their hop count at least once. Furthermore, all nodes will eventually know their correct hop count: Assume that a node A received the hop count $k + 1$. If a route of k hops from the gateway to A exists, it will receive a message of $k - 1$ from a lower count neighbour, causing A to update its hop count to k. It will then proceed to inform nodes further away from the gateway of the lower hop count, prompting them to update in the same manner.

```
1 onReceiveMessage(Message m) {
2   if (m.type == SensorType &&
3       m.hopCount > this.hopCount) {
4     sendMessage(SensorType, this.hopCount,
5       m.data);
6   }
7 }
```

Listing 2: Sensor Data Retrieval Algorithm

This propagation requires at most $O(\mathcal{N}^2)$ messages: In a network of $\mathcal{N}$ nodes, the highest initial hop counts are the sequential hop counts $0, 1, \ldots, \mathcal{K}$, with the highest hop count $\mathcal{K} = \mathcal{N} - 1$. Whenever a node with hop count k receives a propagation message with hop count $k - 2$, it reduces its hop count by 1 to $k - 1$. In the worst case, this process reduces the hop count in steps of one down to 1, resulting in k messages sent. If this happens for all nodes in the network, the network with $\mathcal{N}$ nodes will send $\sum_{i=1}^{\mathcal{N}-1} i = \frac{\mathcal{N}\cdot(\mathcal{N}-1)}{2} \leq \mathcal{N}^2$ propagation messages.

4.2 Sensor Data Retrieval

Once the hop count is established, a nanonode can send sensor data to the gateway. When it broadcasts its data, it attaches its hop count to the message. Nodes in the network now forward the message according to Listing 2, that is, only if it has been received from a node with a higher hop count. This causes the broadcast to continue only towards the gateway, leaving the remaining network uninvolved.

This broadcast must terminate, as no two nodes exist that both forward each other's messages. Nevertheless, the amount of sent messages can still be very high. If a node n_k with hop count k receives a message from all nodes of hop count $k + 1$, it will forward each one, effectively sending $\mathcal{N}_{k+1}$ messages. If all nodes $\mathcal{N}_k$ with hop count k do so, they send $\mathcal{N}_k \cdot \mathcal{N}_{k+1}$ messages. With the upper bound $\mathcal{N} \geq \mathcal{N}_k$, the message count can be estimated as $\prod_{i=1}^{\mathcal{K}} \mathcal{N}_i \leq \mathcal{N}^{\mathcal{K}}$, it thus requires $O(\mathcal{N}^{\mathcal{K}})$ messages.

4.3 Optimisation: Destructive Retrieval

To reduce the amount of messages caused by a single sensor message, we propose a *destructive retrieval* inspired by the propagation phase. The hop count propagation phase limits message counts implicitly, as each nanonode forwards a message only when its state changes, namely its hop count decreases. During sensor data retrieval, nanonodes can track forwarding similarly by resetting their hop count to ∞. To do so, extend the algorithm in Listing 2 with the line `this.hopCount = ∞;` after line 5.

Now, each node forwards the sensor data message only once, resulting in at most $O(\mathcal{N})$ messages. However, the node's hop counts are lost in the process, and the gateway has to start another propagation phase to reestablish the hop count. The total amount of messages sent per sensor message thus is $O(\mathcal{N} + \mathcal{N}^2) = O(\mathcal{N}^2)$.

Destructive retrieval has several additional aspects worth noticing. Until the hop count is fully reestablished, the network around the gateway cannot forward other messages. This renders message retrieval essentially sequential, which can be prohibitive if the new propagation takes a long time. More subtly, if a nanonode n_k with hop count k sends a message to the gateway, the loss of network structure blocks nanonodes n_r with a higher hop count $r > k$. However, the n_r do not know about the network outage, and will unnecessarily try to send their data anyway.

4.4 Sending Instructions to Nodes

In addition to retrieving sensor data, a network may need to transport commands from the gateway to a set of nodes. This already happens during the propagation phase: The hop count is carried into the network. A gateway may thus simply attach additional command data to the propagation message to send it into the network. If several command messages need to be sent, a destructive broadcast may be employed, where a message is only forwarded by nodes with a hop count smaller than ∞.

As usually not all nodes should be addressed, messages may need to carry an address, for example a Function Centric Networking [14] location, which addresses nodes based on their capabilities and location within a human body. Alternatively, [17] employs several gateways, and thus several hop counts per node, to create a hop count based coordinate system. It addresses nodes based on their hop count distances to these gateways. Both addressing approaches require a node to store additional data about itself, which necessitates a more complex node design.

4.5 Multiple Gateways

Even though Hop Count Routing specifies a single microscale gateway, multiple gateways are possible. If a new gateway is added, it starts with a propagation phase. The hop count propagation message will spread through the network until it reaches equidistant nodes. The network is effectively partitioned, where the boundary nodes can communicate with both gateways. This behaviour differs from [17], where each gateway propagates messages through the whole network, and the nodes store multiple hop counts.

4.6 Mobile and Degrading Nodes

As described in Section 2.2, nanonodes in a real-world scenario will likely move as well as degrade. This section provides an intuition how these challenges can be addressed in future work:

Constantly changing positions and node loss results in a highly dynamic network topology. To maintain a valid network state, gateways may periodically initiate a new hop count establishing phase as described in Section 4.1. To clear the obsolete network state, gateways must send a new message type triggering all nodes to set their hop count to ∞. Similarly, the destructive retrieval algorithm needs to reestablish the hop count topology after each sensor data retrieval. A dynamic network can exploit this property, and combine network structure updates with sensor data retrieval.

Timing is important when establishing a new hop count: If delayed copies of a message still exist in the network, they may happen upon the new hop count, get forwarded to the gateway, and thus block precious network resources.

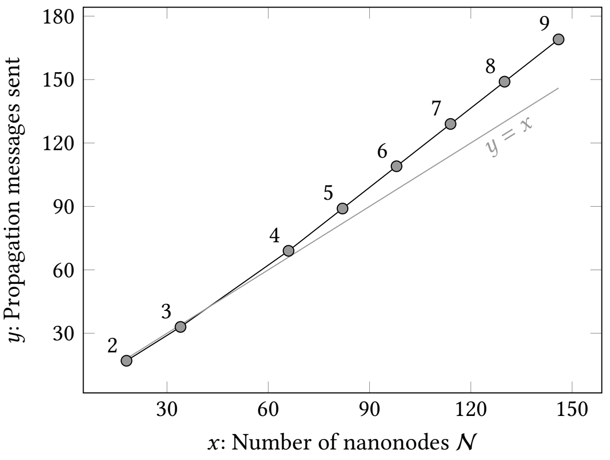

Figure 3: Messages sent during the propagation phase. Numbers show the highest hop count in the respective network.

5 EVALUATION

Sections 4.1 to 4.3 provided a short discussion of the performance that we can expect from the Hop Count Routing algorithm: The propagation phase will require $O(\mathcal{N}^2)$ messages, while the retrieval phase requires naively $O(\mathcal{N}^{\mathcal{K}})$ or destructively $O(\mathcal{N}^2)$ messages.

In order to more precisely estimate of expectable values, we simulate the Hop Count Routing algorithm with the network simulator *ns-3* [12]. The IEEE-P1906.1 reference implementation [1, 3] provides an implementation of electromagnetic communication using terahertz waves.

5.1 Simulation Parameters

The simulated nanoscale environment consists of an empty tube with a diameter of 4 mm and varying length, approximating the shape of a blood vessel. A set of nanonodes is distributed within the vessel, with a gateway n_G at one end, and an observed phenomenon, detected by a sensor nanonode n_S, at the other. The communication distance is 2 mm as explained in Section 1.

The main parameter for network connectivity is the node concentration, that is, the average amount of nodes in the given 3D volume. With a higher concentration, nodes are closer to each other, and can reach more neighbours. Furthermore, the simulation investigates two different placement schemes, namely regular distribution on a grid, and completely random distribution [6].

As described in Section 2.2, the simulation assumes nodes to be indestructible, have infinite energy, not move, and communicate perfectly.

During each run, the simulation performs a single pass of Hop Count Routing, with both a propagation and retrieval phase. The retrieval phase starts at the sensor node n_S, thus needs to travel the whole network. Different simulations compare naive and destructive retrieval algorithms, in regular and random node distributions.

5.2 Simulation Results

Each simulation logs the amount of messages generated during the propagation phase to validate the theoretical assumptions from

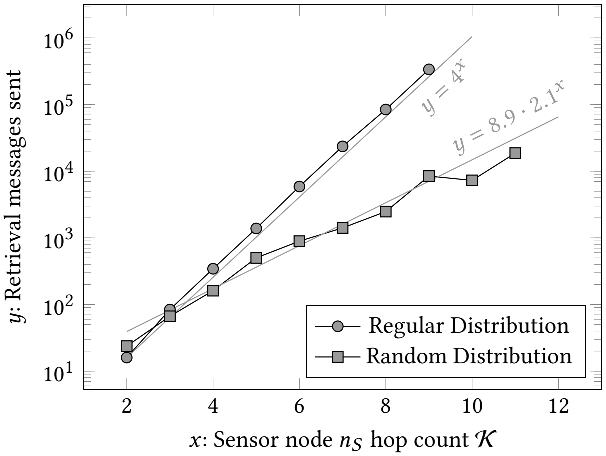

Figure 4: Messages sent for naive retrieval.

Section 4.1. Results in Figure 3 indicate a message count only slightly above the number of nodes, with a maximum of 15% overhead in a network of 146 nodes (169 messages). This suggests that the worst-case scenario of $O(\mathcal{N}^2)$ as given in Section 4.1 is unlikely to occur in an actual deployment.

Figure 4 shows the evaluation of the naive retrieval algorithm given in Section 4.2. Both regular and random distribution observe the same overall node density of $0.729\,\mathrm{mm}^{-3}$. The results confirm the theoretical assumption: The number of messages grows exponentially with the hop count k of n_S, yielding a growth of close to 4^k for a grid-placed network. It is interesting to note the base 4: This is the average number of nodes of a lower hop count that a node in the network can reach directly. Again, this supports the intuition in Section 4.2, which conservatively assumes the overall node count $\mathcal{N}$ as an upper bound for this specific connectivity to lower hop counts.

With a random node distribution, the amount of messages is less linear. For example, for networks of hop count 9, message count ranges from 50 to 47 524. Still, an overall slower increase in messages is visible, which correlates with the lower average direct connectivity to nodes with a lower hop count (2.1 instead of 4). For very small networks however, an random distribution can yield higher numbers of messages, if the nanonodes are placed in a well-connected layout.

Both results for the naive retrieval algorithm show the exponential growth of messages expected in Section 4.2. The destructive retrieval given in Section 4.3 is supposed to require fewer messages, estimated at $O(\mathcal{N})$. Figure 5 shows simulation results for a network of nodes distributed on a regular grid for a destructive retrieval phase. With increasing network size, the number of messages closely approximates the number of nodes $\mathcal{N}$, indicating that nearly all nodes participate in the retrieval. For small networks, the message count is lower, as nodes with the same hop count as the detecting node n_S do not need to forward the message. This effect diminishes with larger networks, resulting in the approximation to $y = x$.

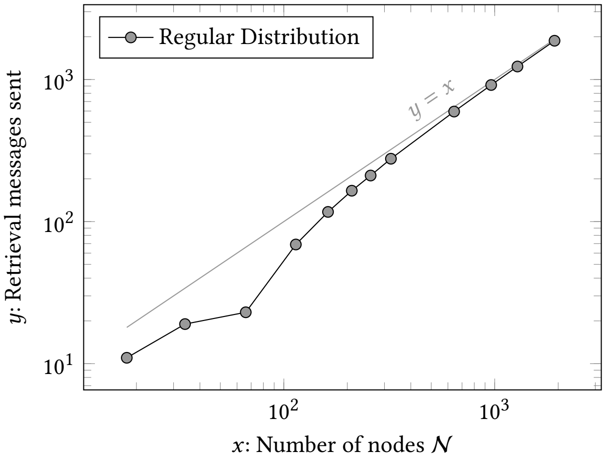

Figure 5: Messages sent during destructive retrieval.

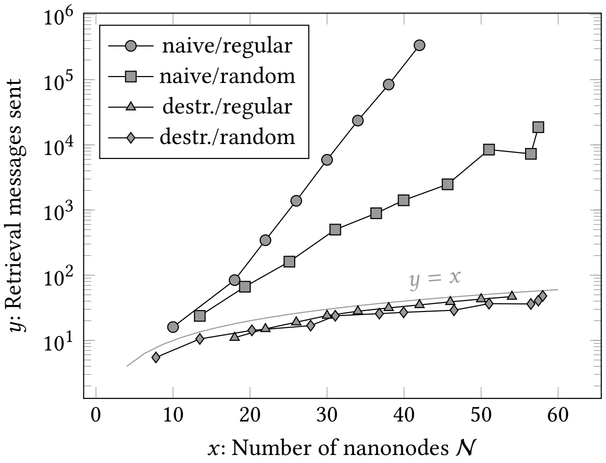

Figure 6: Comparison of retrieval schemes.

5.3 Comparison of Retrieval Algorithms

Figure 6 provides a comparison of both retrieval algorithms, naive and destructive, for regular and random networks. As expected, the identity function $y = x$ neatly separates both algorithms, with the destructive retrieval always requiring less than $\mathcal{N}$ messages, and the naive algorithm requiring considerably more.

This comparison shows message counts for sending only a single message from a sensor to a gateway. To gauge continuous usage, two additional aspects apply:

Firstly, the destructive retrieval will require a new propagation phase after each retrieval. While this may help to mitigate topological changes from a network in motion, it requires more than $\mathcal{N}$ additional messages according to Figure 3. Still, the amount of messages is smaller than in the case of naive retrieval.

Secondly, as already noted in Section 4.3, the destructive retrieval algorithm is inherently sequential, it can only send one message at a time. This can increase packet loss, especially for networks that require parallel message routing. Faster re-establishment of the hop count can help to mitigate packet loss, but the possibility of collisions remains.

5.4 Algorithm Runtime

The runtime of the algorithm is extremely low, with one propagation and one destructive retrieval phase taking about 80 ps for a network with a maximum hop count $\mathcal{K} = 10$ over a distance of 12 mm. This communication speed is very close to the speed of light, as the current nanodevice model includes no processing delay, and the IEEE-P1906.1 reference implementation simulates only propagation delay, but no transmission delay. Further research is required to provide useful results.

5.5 Technical Feasibility

The Hop Count Routing algorithm assumes nanonodes with limited computational capability, where the limiting factors are required memory and the number of distinct operations [8]. Listings 1 and 2 show all required operations, which are *equality* on message types, *comparison* and *increment* of numbers, as well as boolean *and*. Additionally, the nanonodes require memory for one integer to store their own hop count.

In fact, the hop count can be stored as any sufficiently large set of totally ordered symbols. This may render the algorithm feasible for biological nanonodes, which may for example store the hop count as a DNA snippet of a certain length.

6 CONCLUSION

This paper presents a routing approach for medical nanonetworks. We assume a topology comprising one microscale gateway on or in a human's skin and a large amount of nanonodes inside the body, providing sensor information. Due to the limited transmission range, messages are relayed in a multi-hop fashion. As the nanonodes are severely resource constrained, they will be unable to carry unique identifiers or store and maintain complex routing tables.

We propose a routing algorithm using a topology based on the hop count distance from the gateway node. Nodes determine their hop count with a propagation phase, which traverses the whole network, counting the number of hops taken so far. The topology allows to route messages towards the gateway without any additional information. A naive algorithm produces an exponential number of messages. An improvement resets the hop count after transmitting a sensor message, and results in only a linear amount of messages. However, the latter requires to rebuild the hop count topology whenever a message has been sent to the gateway.

In our analytical and simulative evaluation, we simplify some assumptions about the medical nanonetwork, we for example assume nodes to be static. While we can provide some intuitions about how we expect the algorithms to behave when these assumptions are relaxed, further work must inspect actual behaviour. In particular, finite energy restricts communication capabilities, and might have a great impact on the algorithm's performance.

Additionally, we intend to further refine the algorithm itself. Even with destructive retrieval, it requires a possibly large amount of messages to send one higher-level message to the gateway. One approach may use a small amount of additional memory to estimate the amount of nodes with smaller or larger hop counts in the immediate neighbourhood. With this information, we can lower the forwarding frequency, for example using probabilistic forwarding. Still, this will require more complex nanonodes, and can thus be more difficult to deploy on actual hardware.

REFERENCES

[1] IEEE Std. 1906.1-2015. 2016. IEEE Recommended Practice for Nanoscale and Molecular Communication Framework. (Jan 2016), 1–64. https://doi.org/10.1109/IEEESTD.2016.7378262

[2] Ian F. Akyildiz and Josep Miquel Jornet. 2010. The Internet of nano-things. *IEEE Wireless Communications* 17, 6 (December 2010), 58–63. https://doi.org/10.1109/MWC.2010.5675779

[3] Stephen F. Bush, Janet L. Paluh, Guiseppe Piro, Vijay Rao, Venkatesha Prasad, and Andrew Eckford. 2015. Defining Communication at the Bottom. *IEEE Transactions on Molecular, Biological and Multi-Scale Communications* 1, 1 (March 2015), 90–96. https://doi.org/10.1109/TMBMC.2015.2465513

[4] 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

[5] Lester Randolph Ford Jr. and Delbert Ray Fulkerson. 1962. *Flows in Networks.* Princeton University Press.

[6] Regine Geyer, Marc Stelzner, Florian Büther, and Sebastian Ebers. 2018. BloodVoyagerS - Simulation of the work environment of medical nanobots. In *5th ACM International Conference on Nanoscale Computing and Communication 2018 (ACM NanoCom'18).* Reykjavik, Iceland. https://doi.org/10.1145/3233188.3233196 forthcoming.

[7] Robert A. Freitas Jr. 1999. *Nanomedicine, Volume I: Basic Capabilities.* Landes Bioscience, Georgetown, Texas.

[8] Florian-Lennert Adrian Lau, Florian Büther, and Bennet Gerlach. 2017. Computational Requirements for Nano-machines: There is Limited Space at the Bottom. In *Proceedings of the 4th ACM International Conference on Nanoscale Computing and Communication (NanoCom '17).* ACM, New York, NY, USA, Article 11, 6 pages. https://doi.org/10.1145/3109453.3109458

[9] C. Liaskos and A. Tsioliaridou. 2015. A Promise of Realizable, Ultra-Scalable Communications at Nano-Scale:A Multi-Modal Nano-Machine Architecture. *IEEE Trans. Comput.* 64, 5 (May 2015), 1282–1295. https://doi.org/10.1109/TC.2014.2329684

[10] G. Malkin. 1998. *Routing Information Protocol Version 2.* RFC 2453. RFC Editor. 1–39 pages. https://doi.org/10.17487/RFC2453

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

[12] George F. Riley and Thomas R. Henderson. 2010. The *ns-3* Network Simulator. In *Modeling and Tools for Network Simulation.* 15–34. https://doi.org/10.1007/978-3-642-12331-3_2

[13] Patricia Ruiz and Pascal Bouvry. 2015. Survey on Broadcast Algorithms for Mobile Ad Hoc Networks. *Comput. Surveys* 48, 1, Article 8 (July 2015), 35 pages. https://doi.org/10.1145/2786005

[14] Marc Stelzner, Falko Dressler, and Stefan Fischer. 2016. Function Centric Networking: An Approach for Addressing in In-Body Nano Networks. In *Proceedings of the 3rd ACM International Conference on Nanoscale Computing and Communication (NANOCOM'16).* ACM, New York, NY, USA, Article 38, 2 pages. https://doi.org/10.1145/2967446.2967479

[15] 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 (Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering), ICST, Brussels, Belgium, Belgium, 58–64. https://doi.org/10.4108/eai.15-12-2016.2267585

[16] 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

[17] A. Tsioliaridou, C. Liaskos, E. Dedu, and S. Ioannidis. 2017. Packet Routing in 3D Nanonetworks: A Lightweight, Linear-path Scheme. *Nano Communication Networks* 12 (6 2017), 63–71. https://doi.org/10.1016/j.nancom.2017.01.001

[18] Rui Zhang, Ke Yang, Qammer H. Abbasi, Khalid A. Qaraqe, and Akram Alomainy. 2017. Analytical Modelling of the Effect of Noise on the Terahertz In-vivo Communication Channel for Body-centric Nano-networks. *Nano Communication Networks* (2017). https://doi.org/10.1016/j.nancom.2017.04.001