This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TMC.2020.3006535, IEEE Transactions on Mobile Computing

Multi-hop Deflection Routing Algorithm Based on Reinforcement Learning for Energy-Harvesting Nanonetworks

Chao-Chao Wang, *Member, IEEE,* Xin-Wei Yao, *Member, IEEE,* Wan-Liang Wang, and Josep Miquel Jornet, *Member, IEEE*

Abstract—Nanonetworks are composed of interacting nano-nodes, whose size ranges from several hundred cubic nanometers to several cubic micrometers. The extremely constrained computational resources of nano-nodes, the fluctuations in their energy caused by energy harvesting processes, and their very limited transmission range at Terahertz (THz)-band frequencies (0.1-10 THz), make the design of routing protocols in nanonetworks very challenging. A multi-hop deflection routing algorithm based on reinforcement learning (MDR-RL) is proposed in this paper to dynamically and efficiently explore the routing paths during packet transmissions. Firstly, new routing and deflection tables are implemented in nano-nodes, so that nano-nodes can deflect packets to other neighbors when route entries in the routing table are invalid. Secondly, one forward updating scheme and two feedback updating schemes based on reinforcement learning are designed to update the tables, namely, on-policy and off-policy updating schemes. Finally, extensive simulations in networks simulator-3 are conducted to analyze the performance of MDR-RL using different updating policies, as well as to compare the performance with other machine learning routing algorithms based on Neural Networks and Decision Tree. The results show that the MDR-RL can increase the packet delivery ratio and number of delivered packets, and can decrease the packet average hop count.

Index Terms—Nanonetworks, deflection routing, reinforcement learning, energy harvesting, THz communications

1 Introduction

Nanotechnology enables the development of nano-devices, whose size ranges from a few hundred cubic nanometers to a few cubic micrometers. At this scale, the physical properties of new nanomaterials and nanostructures lead to new capabilities and functionalities. For instance, nano-bio-sensors can be utilized to detect biological markers directly in the blood or pathogens in surfaces or in the air with unprecedented accuracy [1], [2]. Nevertheless, due to their reduced size, nano-nodes have very limited computational and memory resources. However, by means of communications, nano-nodes in nanonetworks can overcome their limitations and enable new collaborative applications [3]. Nanonetworks can be utilized in biomedical areas (e.g., advanced health monitoring system [4] and drug delivery system [5]), environmental areas (e.g., agriculture plant monitoring system [6] and pesticide control system [7]) and security areas (e.g., food safety control system [8] and chemical attack prevention [9]).

There are two main alternative communication technologies for nanonetworks, namely, molecular communication and electromagnetic (EM) communication. In this paper, we concentrate on EM-based nanonetworks, where nano-nodes communicate with each other through EM radiation. The very small size of nano-nodes and, thus, of their transceivers and antennas, imposes the utilization of ultra-high transmission frequencies. Among others, the implementation of graphene-based plasmonic transceivers and antennas enables the communication of nano-nodes at Terahertz (THz)-band frequencies, i.e., from 100 GHz to 10 THz [10]. In addition, novel nanomaterials and nanofabrication techniques provide new methods to produce electronic nano-components, including nano-batteries, nano-processors, and nano-memorizers [11], [12]. All the above are driving the EM-based nanonetworks to become true.

In the past few years, several accomplishments for nanonetworks have been achieved in the areas of nano-node design [13], [14], [15], physical [16], [17], [18] and link-layer protocols design [19], [20]. However, only a few papers have studied the routing algorithms. There are several challenges in the design of routing protocols for nanonetworks. In particular,

- The limited transmission range. On the one hand, the THz band suffers from high propagation losses due to both the low effective area of THz nano-antennas as well as molecular absorption [21]. On the other hand, the power of the transmitted signals is limited both by the maximum output power of nano-transceivers and the maximum peak power that nano-batteries can hold [22], [23]. Therefore, the transmission range of nano-nodes is very short and, thus, multi-hop links are needed to deliver packets

- *C. C. Wang is with the College of Mathematics, Physics and Information Engineering, Jiaxing University, Jiaxing, 314001, China. E-mail: ccwang@zjxu.edu.cn*
- *X. W. Yao and W. L. Wang are with the College of Computer Science and Technology, Zhejiang University of Technology, Hangzhou, 310023, China. E-mail: ccwang@zjut.edu.cn, xwyao@zjut.edu.cn, zjutwwl@zjut.edu.cn*
- *J. M. Jornet is with the Department of Electrical & Computer Engineering, Northeastern University, Boston, Massachusetts 02115, USA. E-mail: j.jornet@northeastern.edu*

from sources to destinations, when the distance between them is beyond their transmission ranges.
- The energy fluctuations of nano-nodes. To extend the lifetime of nanonetworks, energy harvesting systems (e.g., triboelectric [24] and piezoelectric [25] nanogenerators) have been proposed to be implemented in nano-nodes. Therefore, the energy of nano-nodes fluctuates non-monotonically in time, which could result in transmission failures. Hence, the routing algorithm needs to update the routing strategies when the energy capacity of a nano-node changes.
- The extremely limited onboard memory/buffer. Nano-nodes might be able to store only one packet due to the limited onboard memory/buffer. Therefore, when a nano-node has a packet to be sent/forwarded, it cannot process packets from other nano-nodes, which leads to immediate packet loss. Hence, the packet store and forward policy in traditional Wireless Sensor Networks (WSNs) might be not suitable for nanonetworks.

As a result of the above three challenges, traditional routing algorithms in wireless networks are not suitable for nanonetworks. In this paper, we propose a multi-hop deflection routing algorithm based on reinforcement learning (MDR-RL), with the following properties:

1) To address the packet loss when the next-hop nano-node in the routing table is not available (due to buffer or energy outages), a deflection table is implemented to direct the packet through second-best routes and prevent packet drop at the buffer.
2) An energy prediction scheme is established to enhance the deflection decision-making process.
3) One feedforward and two feedback update schemes are designed with several nano-node attributes, including deflection ratio, loss probability, hop count to destination and energy status.

We analytically and numerically investigate the performance of the proposed protocol in terms of exploration ability, convergence and overhead. Moreover, extensive simulations in network simulator-3 (ns-3) are conducted to compare our proposed routing algorithm with two other learning-based protocols, namely, Neural Networks Routing (NNR) proposed in [26] and Decision Tree Routing (DTR) proposed in [27]. The results show that our proposed MDR-RL algorithm with on-policy updating scheme has the best performance comprehensively.

The remainder of this paper is organized as follows. In Sec. 2, the related work is introduced. The updating processes of MDR-RL are presented in Sec. 3, including one feedforward and two feedback updating schemes. In Sec. 4, the detailed operations of MDR-RL are presented. Simulations and analyses are provided in Sec. 5. Finally, we conclude the paper in Sec. 6.

2 Related Work

In this section, we describe and discuss existing routing algorithms for nanonetworks as well as deflection routing algorithms developed for traditional networks.

2.1 Existing Routing Algorithms in Nanonetworks

In the related literature of nanonetworks, a few contributions have been made to develop appropriate routing algorithms. The majority of the existing solutions are based on the flooding algorithm. In such protocols [28], a nano-node broadcasts its packet, all other nano-nodes receive the packet and help to forward it to the next-hop nano-node (ultimately, if a path between the source node and the destination node exists, the flooding algorithm will also follow that, among all the other paths). Although flood-based routing protocols are simple and robust, it requires many energy and memory resources and, thus, reduces the performance of nanonetwork. Recently, a few protocols have been proposed to improve the energy efficiency of flooding-based strategies by restricting the flooding area, including a COordinate ROuting Algorithm for 2D ad-hoc nanonetwork proposed in [29], the DEployable ROUting system (DEROUS) proposed in [30], and the Stateless Linear Routing (SLR) proposed in [31].

While these are steps in the right direction, the above three routing algorithms are all flood-based, which lead to high energy consumption, and a special network structure to assign the coordinates. Additionally, energy and memory issues are not considered in these routing algorithms.

2.2 Deflection Routing in Other Networks

Despite being fundamentally different, optical burst switching (OBS) networks [32], [33], [34] and electromagnetic nanonetworks share one aspect in common, namely, the very limited memory or bufferless operation [35]. In OBS, as opposed to the traditional routing policies based on storing a packet and forwarding to the best route when available, deflection routing has been proposed. Deflection routing is a rerouting technology that can eliminate heavy data traffic and avoid packet buffering when network packets competition happens and nodes possess little to no buffer. In this case, the packets are deflected away from the main path by the routing algorithm rather than discarded. A combined probabilistic deflection and retransmission scheme is presented in [36] for buffer-less OBS networks. In the algorithm, when more than two bursts come to one node, only one of them can choose the shortest port, however, the other bursts choose other idle ports with a probability.

The deflection routing algorithm also has been used in on-chip communication networks. In [37], an energy-efficient deflection routing algorithm is proposed, in which a multi-channel network interface and a novel deflection routing based on the turn model are designed. The proposed algorithm cannot only improve the packet delivery probability, but also reduces the hardware cost and energy consumption.

In nanonetworks, due to the energy and buffer constraints, the packets could be easily dropped when congestions occur. Hence, deflection routing can be a viable routing solution in nanonetworks. However, due to the significant differences between OBS networks and energy-harvesting nanonetworks, the deflection routing algorithms should be redesigned for nanonetworks. Especially, the dynamic of traffic load and fluctuations of energy capacity of nano-nodes should be considered.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TMC.2020.3006535, IEEE Transactions on Mobile Computing

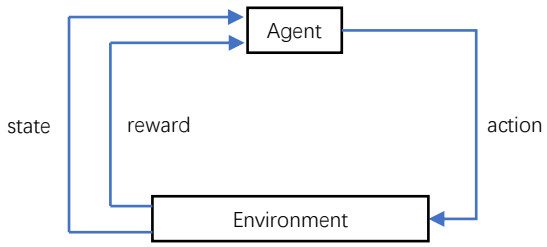


Fig. 1. The framing of reinforcement learning.

2.3 Reinforcement Learning

Reinforcement learning is a kind of machine learning technique, which emphasizes how software agents act based on the environment to maximize the expected benefits. Fig. 1 presents a typical scenario of reinforcement learning. The agent takes actions, and the environment takes the status of the agent and its action as inputs, and returns the reward and state as outputs. The reward refers to the feedback that the algorithm uses to measure the success or failure of the agent's action. Then, the agent can update its parameters by the reward and new state so that better decisions become more likely to be made in the future [38]. Besides, the value function is also a key component of RL systems, which estimates "how good" it is to be in a given state.

In [39], a state-action-reward-state-action (Sarsa) routing algorithm combined with a fuzzy logic system was proposed for mobile ad-hoc networks. The residual energy and energy drain rate of nodes are utilized to update the routing table. To address the local optimum issue, a selection probability scheme was introduced, in which the best routing path has the highest selection probability, while the routing path with low probability can still be selected. Nevertheless, the routing algorithm proposed in [39] is implemented based on the Ad hoc On-Demand Distance Vector Routing (AODV) protocol. Therefore, it is hard to be realized in nanonetworks. Moreover, the traffic load, hop count and packet loss probability are not considered in [39], which cannot be neglected in nanonetworks.

For our early study, a deflection routing algorithm based on Q-learning for nanonetworks, which used a forward updating scheme to update the routing and deflection tables, can be found in [40]. As the best of our knowledge, this is the first time that RL based deflection routing is proposed for nanonetworks to overcome memory issues.

3 The Updating Processes of MDR-RL

The main objective of MDR-RL is to adaptively explore appropriate routing paths to transmit packets by taking into account the dynamics of nanonetworks. In this section, the details of the updating processes of MDR-RL are presented. First, the structure of the deflection and routing tables, the deflection scheme and the energy prediction scheme are presented. Then, one forward updating policy and two feedback updating policies are presented. The associated notations in this section are listed in Table 1.

3.1 Tables Structures

In the MDR-RL, a routing table is designed to route packets, and a deflection table is established to deflect packets when routing paths in the routing table are invalid due to energy or memory issues. The tables are empty initially, and are built up during transmissions. Both can be updated based on the updating process of MDR-RL, which is described in Secs. 3.4 and 3.5.

TABLE 1
Notations of the updating processes of MDR-RL in Sec. 3

Symbols (examples)	Description
d, x, y, z, s	Nano-nodes
$Q_{z_1}(d_1, y_j)$	Q-value from z_1 to destination d_1 via y_j
$R_{z_1}(d_1, y_j)$	Recovery rate from z_1 to destination d via y_j
$H_{z_1}(d_1, y_j)$	Hop Count from z_1 to destination d_1 via y_j
$T_{z_1}(d_1, y_j)$	Time when this route entry is updated
E_{max}	Maximum battery capacity of nano-node
$E_{y_j}^{z_1\prime}$	Predicated energy status of y_j in z_1
$\Delta t_{z_1}(y_j)$	Time duration between the time when the route entry was updated and current time
t_c	Current time
$S_{y_j}^h$	Energy harvesting rate of y_j
$S_{y_j}^c$	Energy consuming rate of y_j
ω	Predicted energy
$r_{y_1}(d_1, x_1)$	Reward information from y_1
$p_{def}^{y_1}$	Deflection ratio of y_1
$p_{loss}^{y_1}$	Packet loss probability of y_1
$N_{def}^{y_1}$	Number of deflection times of y_1
$N_{def}^{y_1}$	Number of lost packets of y_1
$N_{trans}^{y_1}$	Number of transmitted packets of y_1
$C_{energy}^{y_1}$	Consumed energy percentage of y_1
$E_c^{y_1}$	Residual energy of y_1
α	Learning rate
β	Recovery coefficient
γ	Decay coefficient
ϕ	Difference between reward and Q-value

We consider an arbitrary nanonetwork with several nano-nodes (see in Fig. 2). Each nano-node implements the MDR-RL, and has both routing and deflection tables. When a nano-node generates or receives a packet, it investigates the routing table to find the next-hop nano-node. For a destination, there exists only one route entry. The composition of each route entry is as follows:

- *Destination nano-node ID*
- *Next-hop nano-node ID*
- *Q-value* to destination nano-node via next-hop nano-node
- *Recovery Rate* to destination nano-node via next-hop nano-node
- *Hop Count* to destination nano-node
- *Time* when this route entry is updated
- *Route Valid Flag*
- *Lifetime*

where *Q-value* refers to the weight of the corresponding routing path. The higher the Q-value is, the more resources (energy, buffer, and hops) are consumed by the routing path, i.e., the Q-value is an indicator of the path quality. The *Recovery Rate* is applied to estimate the rate that the nano-node recovers to its original status. For instance, nano-nodes without energy can recover by harvesting energy from the environment. The *Route Valid Flag* identifies whether the

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TMC.2020.3006535, IEEE Transactions on Mobile Computing

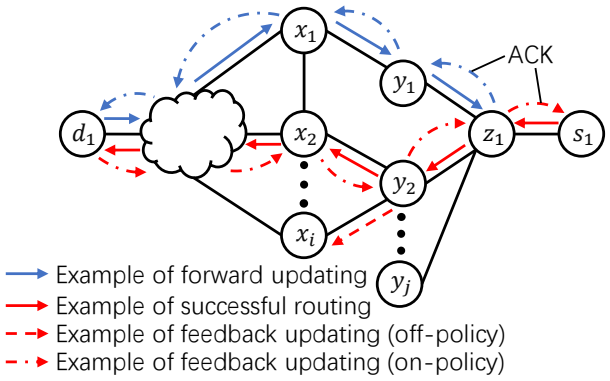


Fig. 2. An arbitrary nanonetwork

route entry is valid or not. It is enabled when the nano-node receives a data packet or ACKnowledgment (ACK) from the corresponding nano-node, and is disabled when the nano-node receives a Negative ACKnowledgment (NACK) or meets a timeout. The *Lifetime* refers to how long the route entries survive in the routing table. Initially, the routing table is empty, and route entries are added and updated during packet transmissions period.

Because the size of the nano-nodes is very small, their energy capacity and memory/buffer are extremely limited. Therefore, route entries may become invalid when one of the following three things happens: (i) the next-hop nano-node in the routing table runs out of energy; (ii) the next-hop nano-node receives or generates packets to be forwarded/sent and, thus, become busy communicating with another nano-node, and has no more idle memory/buffer to store new packets; (iii) errors occur during the transmissions, which may be caused by channel congestion, modulation errors, and so on. All the above problems lead to the failure of packet transmission. To address these issues, the deflection table is introduced so that the transmitting nano-nodes can deflect packets to other nano-nodes if the next-hop nano-node in the routing table is invalid. In the proposed deflection table of a nano-node, every entry is indexed by destinations and its neighbors. Hence, there may exist multi deflection route entries for one destination. For instance, as shown in Fig. 2, if z_1 wants to send a packet to d_1 and the next-hop nano-node in the routing table is invalid, it has several alternative nano-nodes (from y_1 to y_j) to deflect the packet. Continuing with the example of the arbitrary nano-node z_1, the composition of each deflection route entry is as follows:

- $Q_{z_1}(d_1, y_j)$ - Q-value from z_1 to destination d_1 via y_j
- $R_{z_1}(d_1, y_j)$ - Recovery rate from z_1 to destination d via y_j
- $H_{z_1}(d_1, y_j)$ - Hop Count from z_1 to destination d_1 via y_j
- $T_{z_1}(d_1, y_j)$ - Time when this route entry is updated

Consistently with the routing table, the deflection table is empty initially, and is updated during the packets transmissions period.

3.2 Deflection Scheme

Consider that z_1 receives a packet from s_1, and the destination is d_1 (red line in Fig. 2). First of all, z_1 searches for the next-hop nano-node in the routing table, if the next-hop nano-node is invalid, z_1 chooses a neighboring nano-node y_m ($m \in (1, 2, \cdots, j)$) in the deflection routing table to forward the packet, determined by

$$m = \arg\min \left(Q_{z_1}(d_1, y_j) + \Delta t_{z_1}(y_j) R_{z_1}(d_1, y_j) \left(E_{\max} - E_{y_j}^{z_1\prime}\right)\right), \quad (1)$$

where the function of arg min is to obtain the index of the next-hop nano-node with minimum Q-value. y_j is the set of neighboring nano-nodes of z_1, except for the invalid nano-nodes. $R_{z_1}(d_1, y_j)$ is the recovery rate from z_1 to the destination d via y_j. $\Delta t_{z_1}(y_j)$ is the time duration between the time when the route entry was updated and current time t_c, which can be obtained by:

$$\Delta t_{z_1}(y_j) = t_c - T_{z_1}(d_1, y_j), \quad (2)$$

where $E_{\max}$ is the maximum battery capacity of nano-node, $E_{y_j}^{z_1\prime}$ is the predicated energy status of y_j in z_1, which we explain in the following section.

3.3 Energy Prediction Scheme

Due to the very limited battery capacity of nano-nodes, an energy harvesting nano-system is used to extend the lifetime of nanonetworks. However, these lead to frequent energy fluctuations in nano-node (energy is no longer a monotonically decreasing parameter), which can also cause transmission failures. Therefore, an energy prediction scheme is established to select the next-hop nano-node with maximum energy to avoid no energy issues. In the MDR-RL, nano-nodes can share their energy status and energy consumption and harvesting rates during the transmissions. Hence, continue with the example of z_1, it can predict the energy status of neighboring nano-nodes as:

$$E_{y_j}^{z_1\prime} = \begin{cases} E_{\max} & \omega > E_{\max} \\ \left(S_{y_j}^h - S_{y_j}^c\right)\Delta t_{y_i} + E_{y_j}^{z_1} & 0 < \omega < E_{\max} \\ 0 & \omega < 0, \end{cases} \quad (3)$$

where $S_{y_j}^h$ and $S_{y_j}^c$ are the energy harvesting and consumption rate of y_j, respectively. $E_{y_j}^{z_1}$ is the energy status of y_j recorded in z_1 and is updated after last transmission, $\omega = \left(S_{y_j}^h - S_{y_j}^c\right)\Delta t_{y_i} + E_{y_j}^{z_1}$.

3.4 Forward Updating Scheme

The deflection table is updated when the nano-node receives a forwarded packet from others. For instance, z_1 receives a packet from y_1, whose source is d_1 and forwarded through x_1 (blue line in Fig. 2). Since bigger hop count to the destination, higher deflection ratio and loss probability, and higher energy consumption of nano-nodes is more likely to lead to worse routing decisions, we define the reward function as follows:

$$r_{y_1}(d_1, x_1) = Q_{y_1}(d_1, x_1) \cdot \left(H_{y_1}(d_1, x_1) + 1\right) \cdot P_{def}^{y_1} \cdot P_{loss}^{y_1} \cdot C_{energy}^{y_1}, \quad (4)$$

where $Q_{y_1}(d_1, x_1)$ and $H_{y_1}(d_1, x_1)$ are the Q-value and hop count of the corresponding routing path, respectively. $P_{def}^{y_1}$ is the deflection ratio of y_1, and can be expressed as follows:

$$P_{def}^{y_1} = \frac{N_{def}^{y_1}}{N_{trans}^{y_1}}, \tag{5}$$

where $N_{def}^{y_1}$ refers to the time that deflection happens in y_1. $N_{trans}^{y_1}$ refers to the packet quantity transmitted by y_1. $P_{loss}^{y_1}$ is the packet loss probability of y_1, given by

$$P_{loss}^{y_1} = \frac{N_{loss}^{y_1}}{N_{trans}^{y_1}}, \tag{6}$$

where $N_{loss}^{y_1}$ refers to the lost packets quantity. $C_{energy}^{y_1}$ is consumed energy percentage of y_1, which can be expressed as

$$C_{energy}^{y_1} = \frac{E_{\max} - E_c^{y_1}}{E_{\max}} \times 100\%, \tag{7}$$

where $E_c^{y_1}$ is the residual energy percentage of y_1.

In this case, when z_1 receives a packet from y_1, it obtains the reward information $r_{y_1}(d_1, x_1)$ from the packet header and updates its Q-value of corresponding deflection route entry as

$$Q_{z_1}(d_1, y_1) = Q_{z_1}(d_1, y_1) + \alpha \left(\frac{r_{y_1}(d_1, x_1)}{H_{z_1}^r(d_1)} - Q_{z_1}(d_1, y_1) \right), \tag{8}$$

where $H_{z_1}^r(d_1)$ represents the hop count from z_1 to destination d_1 in the routing table, $\alpha\ (0 < \alpha \leq 1)$ represents the *learning rate*, i.e., "how much information" a nano-node learns from the reward of the routing. Then, the recovery rate $R_{z_1}(d_1, y_1)$ is updated as follows:

$$R_{z_1}(d_1, y_1) = \begin{cases} R_{z_1}(d_1, y_1) + \beta \frac{\phi}{\Delta t_{z_1}(y_1)}, & \phi < 0 \\ \gamma R_{z_1}(d_1, y_1), & \phi \geq 0, \end{cases} \tag{9}$$

where $\beta\ (0 < \beta \leq 1)$ refers to the *recovery coefficient*, which adapts to the nano-node energy recovery and network traffic load; and $\gamma\ (0 < \gamma \leq 1)$ refers to the *decay coefficient*, which adapts to the nano-node energy consumption and increase of network traffic load; and $\Delta t_{z_1}(y_1)$ is the time interval between last table updating time and the current time t_c; and ϕ can be expressed as:

$$\phi = \frac{r_{y_1}(d_1, y_1)}{H_{z_1}^r(d_1)} - Q_{z_1}(d_1, y_1). \tag{10}$$

Furthermore, $H_{z_1}(d_1, y_1)$ is updated by the hop count stored in the header of the packet. Finally, the *Time* of the router entry is updated to the current time as:

$$T_{z_1}(d_1, y_1) = t_c. \tag{11}$$

After updating the deflection table, by comparing $Q_{z_1}^r + \Delta t_{y_1} R_{z_1}^r$ and $Q_{z_1}(d_1, y_1) + \Delta t_{y_1} R_{z_1}(d_1, y_1)$, a nano-node judges whether the route entry with the same destination should be replaced by the deflection entry or not, where $Q_{z_1}^r$ and $R_{z_1}^r$ are the Q-value and recovery rate of the corresponding route entry, respectively. If the former is greater, i.e., the path in the routing table consumes more resources than the deflection entry, the route entry is replaced by the deflection entry.

3.5 Feedback Updating Scheme

In MDR-RL, a simple ALOHA type Medium Access Control (MAC) protocol with ACK and NACK is considered to be operating in all nano-nodes. To make full use of the information in the acknowledgment and enhance the learning and updating processes, two feedback updating schemes are presented. Consider that z_1 sends a packet to d_1 by y_2. When y_2 receives the packet, it sends an acknowledgment packet containing the updating information back to z_1, then z_1 can use the updating information to update its routing and deflection tables. In this paper, two different feedback updating schemes are introduced: on-policy updating scheme and off-policy updating scheme. In the on-policy updating scheme, nano-nodes send back the reward of the routing path carried out by the routing or deflection scheme. In the off-policy updating scheme, nano-nodes send back the reward of the routing path with minimum Q-value independently of routing and deflection actions.

3.5.1 On-policy Updating Scheme

Consider that z_1 sends a packet to d_1 by y_2, and y_2 chooses x_2 as the next-hop nano-node according to the routing or deflection scheme (red dot-dash line in Fig. 2). Then y_2 sends an ACK packet back to z_1, which contains the reward information obtained by:

$$\begin{aligned} r_{y_2}(d_1, x_2) =& Q_{y_2}(d_1, x_2) \cdot \left(H_{y_2}(d_1, x_2) + 1 \right) \cdot \\ & P_{def}^{y_2} \cdot P_{loss}^{y_2} \cdot C_{energy}^{y_2}. \end{aligned} \tag{12}$$

After receiving the ACK packet, z_1 can use this reward to update the routing and deflection tables.

3.5.2 Off-policy Updating Scheme

Be different from the on-policy updating scheme, in off-policy updating scheme, when y_2 receives the packet from z_1, it sends back the reward of the routing path with minimum Q-value, which can be decided by:

$$x_k = \arg\min \left(Q_{y_2}(d_1, x_i) + \Delta t_{y_2}(x_i) R_{y_2}(d_1, x_i) \right), \tag{13}$$

where k is the index of the nano-node with minimum Q-value (red dash line in Fig. 2). y_2 sends the ACK with reward information back to z_1 regardless of whether x_k is the next-hop nano-node chosen according to the routing table or deflection schemes. Then, the reward can be expressed as

$$\begin{aligned} r_{y_2}(d_1, x_k) =& Q_{y_2}(d_1, x_k) \cdot \left(H_{y_2}(d_1, x_k) + 1 \right) \cdot \\ & P_{def}^{y_2} \cdot P_{loss}^{y_2} \cdot C_{energy}^{y_2}. \end{aligned} \tag{14}$$

From the differences between the two feedback updating schemes, the off-policy updating scheme is more greedy, which always selects the reward of the routing path with the minimum Q-value. Instead, the on-policy updating scheme uses the reward information of the routing path chosen by the routing or deflection scheme, which helps to transmit the packet successfully, hence, it is "safer" than off-policy updating scheme.

4 The Operations of MDR-RL

In this section, the sending/forwarding and receiving operations of MDR-RL are presented in detail, respectively. Moreover, the exploration, convergence, and overhead of the algorithm are analyzed.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TMC.2020.3006535, IEEE Transactions on Mobile Computing

4.1 Operations of Sending/Forwarding a Packet

The details of sending/forwarding operations are presented in Algorithm 1. The operations of sending and forwarding a packet are similar. When a nano-node generates a packet itself or receives one from others, it first checks whether the energy is sufficient or not. If yes, it investigates the routing table for the next-hop nano-node to the destination. If the corresponding route entry is not available, the transmitting nano-node then looks up the deflection table to choose a deflection path. If there is no available route entry or the corresponding route entry to the destination does not exist, the transmitting nano-node picks one of the neighboring nano-nodes randomly to send/forward the packet. If there is no neighboring nano-node, it drops the packet. The algorithm ends when the nano-node receives an ACK packet from next-hop nano-node or exceeds the maximum deflection time. If the algorithm ends due to exceeding the maximum deflection time, the packet is discarded.

Algorithm 1 Sending/Forwarding operations of MDR-RL

Input: Generate a packet itself or receive a packet from another nano-node

Send/Forward the packet to the next-hop nano-node :

if energy is enough to send the packet **then**
  Look up the routing table;
  if the corresponding route entry in the routing table to the destination is accessible **then**
    Choose the corresponding next-hop nano-node;
  else if the corresponding route entry in the routing table to destination is not accessible **then**
    while deflection time < maximum deflection time and ACK packet is not received from the next-hop **do**
      Set the corresponding route entry invalid;
      if energy is enough to deflect the packet **then**
        Look up the deflection table;
        if deflection entries to the destination exist **then**
          Choose the appropriate deflection nano-node as the next-hop by the deflection scheme;
        else if no deflection entry to the destination exists **then**
          Pick one of the neighboring nano-nodes randomly as the next-hop;
        end if
        Update the reward information of the chosen nano-node into the packet;
        Send/Forward the packet to the corresponding next-hop nano-node;
        if there is no neighboring nano-node **then**
          Drop the packet;
        end if
      else
        Wait for a Round-Trip-Time (RTT) and harvest energy;
      end if
    end while
  end if
end if

4.2 Operations of Receiving a Packet

In MDR-RL, a nano-node uses forwarded data packets and feedback acknowledgment packets to update its routing and deflection tables. As described in Algorithm 2, if the energy of a nano-node is not enough to receive a packet, it discards the packet and tries to send a NACK packet back. When a nano-node receives a packet sent by another nano-node successfully, it extracts the update information from the packet header. When receiving a data packet, a nano-node checks the source address of the packet as the destination in the routing and deflection tables. If the source address in the data packet does not match any route entry in the routing and deflection tables, it adds the corresponding routing path to the tables; otherwise, the matched entry is updated by the reward. When receiving an ACK packet, the corresponding routing or deflection entry in the tables is updated by the updating scheme proposed in Sec. 3. For receiving a NACK packet, the nano-node discards the packet.

4.3 Exploration, Convergence and Overhead Analysis

In this section, the exploration ability, convergence rate and overhead of the MDR-RL are analyzed. The associated notations in this section are listed in Table 2.

4.3.1 Exploration ability

The aim of MDR-RL is to explore the appropriate routing path to transmit packets. However, network traffic load and energy status of nano-nodes may change by time, hence, the routing algorithm should be adaptive to the change. In the MDR-RL, once a nano-node receives forwarded or feedback packets, it updates the Q-value of the corresponding routing and deflection paths by taking advantage of the update information (reward information and basic path information) contained in the packets. The reward information is measured by the Q-value, hop count, the deflection ratio, loss probability and energy status of nano-nodes of the corresponding routing path.

In the routing path updating process, three parameters are utilized to help explore the optimal routing path. The *learning rate* decides how much a nano-node learns from the reward. The *recovery* and *decay coefficients* are designed to adapt to the changes of nano-node energy status and network traffic load. Moreover, two different feedback updating schemes are designed to address different performance requirements.

4.3.2 Convergence of the algorithm

In every round of updating of the MDR-RL, the agent observes the nano-node status s_n, and takes action a_n, observes the subsequent status s_{n+1}, and then obtains the reward r. Hence, (8) can be abstracted and simplified to the following equations:

$$Q_n(s,a) = \begin{cases} (1-\alpha)\, Q_{n-1}(s,a) + \alpha r & \text{if } s = s_n \text{ and } a = a_n \\ Q_{n-1(s,a)} & \text{otherwise.} \end{cases} \quad (15)$$

The above model follows the rules of the action-reply process (ARP), which is an artificially controlled Markov process described in [41]. Hence, according to [41], given the bounded reward r, *learning rate* $\alpha\,(0 < \alpha \leq 1)$, and

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TMC.2020.3006535, IEEE Transactions on Mobile Computing

Algorithm 2 Receiving operations of MDR-RL

Input: Receive a packet from another nano-node

Update the corresponding entry in tables :

1: **if** energy is enough to receive the packet **then**
2: Extract the route information from the header;
3: **if** the receiver is me **then**
4: **if** the packet is a data packet **then**
5: Send back an ACK packet contained with updating information (Off-policy);
6: **if** there is no entry in routing and deflection tables match the source address contained in the packet header. **then**
7: **if** the routing table or deflection table is full **then**
8: Delete the oldest route entry;
9: **end if**
10: Extract the reward information from the header;
11: Add a new route to the routing table and deflection table;
12: **else**
13: Update the routing and deflection tables by the updating scheme;
14: **end if**
15: **if** the destination is me **then**
16: Process this packet;
17: **else**
18: Forward the packet by following the forwarding operations;
19: Send back an ACK packet contained with updating information of the next-hop nano-node chosen by the forwarding operations (On-policy);
20: **end if**
21: **end if**
22: **if** the packet is an ACK packet **then**
23: Update the routing and deflection tables;
24: **end if**
25: **if** the packet is a NACK packet **then**
26: Drop the data packet;
27: **end if**
28: **end if**
29: **else**
30: Drop the packet;
31: **if** energy is enough to send the NACK packet **then**
32: Send a NACK packet back;
33: **end if**
34: **end if**

$\sum_{i=1}^{\infty} {\alpha_n}^i (s,a) = \infty$, $\sum_{i=1}^{\infty} \left[{\alpha_n}^i (s,a)\right]^2 < \infty$, then $Q_{n+1}(s,a)$ converges to $Q^*(s,a)$, i.e.,

$$Q_{n+1}(s,a) \to Q_n^*(s,a) \to \infty, \forall s, a, \tag{16}$$

with probability 1, where $Q_n^*(s,a)$ is optimal action values.

Then, according to the theorem proposed in [42], the convergence rate of MDR-RL can be obtained by

$$|Q_n(s,a) - Q^*(s,a)| \le \frac{B}{n^R}, \tag{17}$$

TABLE 2
Notations of the exploration, convergence and overhead analysis of MDR-RL in Sec. 4.3

Symbols (examples)	Description
s_n	Nano-nodes status
a_n	Actions
a_n	Actions
$Q_n(s,a)$	Action value with status s and action a
B	A suitable constant
$p(s,a)$	Sampling probability of (s,a)
p_{suc}	Probability of transmitting a packet successfully from one nano-node to the next-hop
p_{Data}, p_{ACK}	Probabilities of no error during data and ACK packets transmission
p_{col}	Probability of collision
E_{suc}	Energy consumption of transmitting a packet successfully from one nano-node to the next-hop (no error)
E_1, E_2	Energy consumptions when data and ACK packets are not received correctly
E_{Data}^T, E_{Data}^R	Energy consumption of transmitting and receiving a data packet
E_{ACK}^T, E_{ACK}^R	Energy consumption of transmitting and receiving an ACK packet
B_{Data}, B_{ACK}	The sizes of data and ACK packets
η_{Data}, η_{ACK}	The ratios of symbol "1" in data and ACK packets
B_{Data}, B_{ACK}	The sizes of data and ACK packets
B_{Data}, B_{ACK}	The sizes of data and ACK packets
E_p^T, E_p^R	Transmitting and receiving pulse energy
E_{Trans}, E_{Rece}	Energy consumption of transmitting and receiving a packet successfully from one nano-node to the next-hop (with errors)

and

$$|Q_n(s,a) - Q^*(s,a)| \le B\sqrt{\frac{\log\log n}{n}}, \tag{18}$$

where $B > 0$ is a suitable constant, $R = p_{\min}/p_{\max}$, with $p_{\min} = \min_{(s,a)} p(s,a)$ and $p_{\max} = \max_{(s,a)} p(s,a)$, where $p(s,a)$ is the sampling probability of (s,a).

However, given the dynamic nature of nanonetworks, with frequency changes in traffic load and energy status of each nano-node, as well as the time needed for the network to learn the new scenario, it is possible that the algorithm that does not converge every time. Still, the utilization of routing and deflection tables should minimize the consequences of inaccurate routing information.

4.3.3 Overhead of transmitted packet and updating process

After sending packets, nano-nodes receive the ACK packet from the next-hop nano-node. By referring to [43], the probability of transmitting the packet successfully from one nano-node to the next-hop can be expressed as

$$p_{suc} = p_{Data} p_{ACK} (1 - p_{col}), \tag{19}$$

where p_{Data} and p_{ACK} are the probabilities of no error during data and ACK packets transmission, respectively. p_{col} is the probability of collision.

The transmission could fail when: (i) the data packet is not received correctly by the next-hop nano-node; or (ii) the ACK packet is not received correctly by the sending

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TMC.2020.3006535, IEEE Transactions on Mobile Computing

nano-node. The probability of the above two cases can be expressed as:

$$p_1 = 1 - p_{Data} + p_{Data}p_{col}, \tag{20}$$

$$p_2 = p_{Data}\left(1 - p_{col}\right)\left(1 - p_{ACK}\right). \tag{21}$$

The energy consumption of the above three cases are

$$E_{suc} = E^T_{Data} + E^R_{ACK}, \tag{22}$$

$$E_1 = E^T_{Data}, \tag{23}$$

$$E_2 = E^T_{Data} + E^R_{ACK}, \tag{24}$$

where $E^T_{Data} = \eta_{Data}B_{Data}E^T_p$ is the energy of sending a data packet, η_{Data} is the ratio of symbol "1" in a data packet, B_{Data} is the size of a data packet , E^T_p is the transmitting pulse energy. $E^R_{ACK} = \eta_{ACK}B_{ACK}E^R_p$ is the energy of receiving an ACK packet, where η_{ACK} is the ratio of symbol "1" in an ACK packet, B_{ACK} is the size of ACK, E^R_p is receiving pulse energy. Then, the energy consumption of transmitting a packet successfully from one nano-node to the next-hop can be obtained as follows [19], [43]:

$$E_{Trans} = \frac{1}{p_{suc}}\left(E_{suc}p_{suc} + E_1p_1 + E_2p_2\right). \tag{25}$$

Substituting (19), (20), (21), (22), (23) and (24) into (25), we obtain

$$\begin{aligned} E_{Trans} =& \frac{1}{p_{Data}p_{ACK}\left(1 - p_{col}\right)}\left(E^T_{Data}\left(1 - p_{Data} + p_{Data}p_{col}\right)\right. \\ &\left. + \left(E^T_{Data} + E^R_{ACK}\right)p_{Data}\left(1 - p_{col}\right)\left(1 - p_{ACK}\right)\right) \\ &+ E^T_{Data} + E^R_{ACK}. \end{aligned} \tag{26}$$

Similarly, the energy consumption of receiving a packet can be expressed as

$$\begin{aligned} E_{Rece} =& \frac{1}{p_{Data}p_{ACK}\left(1 - p_{col}\right)}\left(E^R_{Data}\left(1 - p_{Data} + p_{Data}p_{col}\right)\right. \\ &\left. + \left(E^R_{Data} + E^T_{ACK}\right)p_{Data}\left(1 - p_{col}\right)\left(1 - p_{ACK}\right)\right) \\ &+ E^R_{Data} + E^T_{ACK}, \end{aligned} \tag{27}$$

where $E^R_{Data} = \eta_{Data}B_{Data}E^R_p$ is the energy consumption of receiving a data packet. $E^T_{ACK} = \eta_{ACK}B_{ACK}E^T_p$ is the energy consumption of transmitting an ACK packet. Since the hops of different routings could be different, we only consider the situation of one nano-node to the next-hop nano-node, the end-to-end probability and energy consumption can be obtained by multiplying the node-to-node probabilities and summing up the node-to-node energy consumption from hops to hops.

4.3.4 Packets structure

To make the tables adapt to the changes in the nano-node energy status and network traffic load, the forward and feedback packet needs to contain the reward information. Hence, the format of the data packet should be revised. The structure of the data packet (Fig. 3) consists of three parts, including the MAC layer header, the network layer header, and the payload. The *Receiver Address* in the MAC header is the nano-node that the packet is sent to and the *Transmitter Address* identifies the local nano-node. The *Protocol* in the

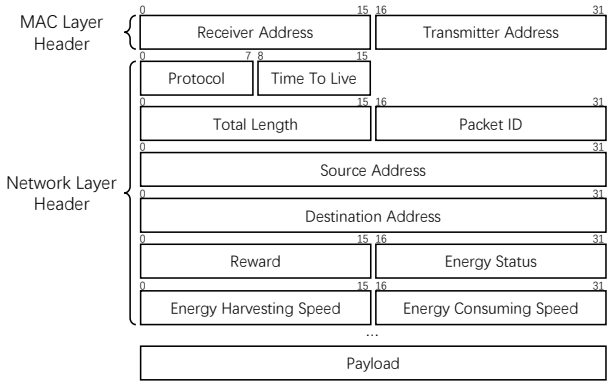


Fig. 3. The structure of a data packet.

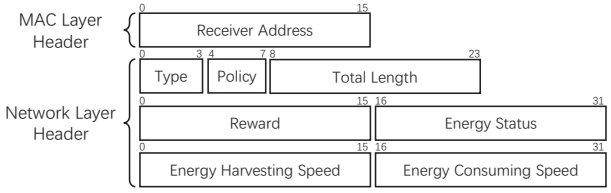


Fig. 4. The structure of a feedback packet.

network layer header decides what kind of routing algorithm the nano-node chooses. The *Time To Live (TTL)* is the number of hops the packet can go through and is utilized to avoid looping in the network. It decreases by one through each hop, and the packet is discarded when the value comes to zero. The *Total Length* is the length of the entire packet. The *Packet ID* is used to identify the packet. *Source Address* and *Destination Address* are the nano-node which generates the packet and the final hop nano-node, respectively. The *Reward*, *Energy Status*, *Energy Harvesting, and Consuming Speed* are used to update the nano-node energy status in both tables. The structure of feedback packet is shown in Fig. 4. There are three types of feedback packet, including ACK, ACK without updating information and NACK. In the NACK packet, there is no reward information. The *Policy* decides which feedback updating scheme is adopted of the nano-node.

Additionally, energy is also limited in nano-nodes. In the MDR-RL, once a nano-node receives a forwarded or feedback packet, it needs to compute the reward to update the routing and deflection tables. Hence, the overhead of this computation process is also considered in the simulations.

5 Simulations and Analysis

In this section, the performance of the proposed MDR-RL is studied with different parameters. Firstly, in Sec. 5.1, three performance metrics in nanonetworks are defined, namely, packet delivery ratio, number of delivered packets and packet average hop count. A nanonetwork environment simulation platform and all the parameters are presented in Sec. 5.2. The actual results are presented in Sec. 5.3 to 5.7.

In our analysis, we compare the MDR-RL using on-policy updating, off-policy updating, no feedback updating, and no updating schemes. For the MDR-RL using no feedback updating scheme, only the forwarded data

TABLE 3
Complexity comparison of different algorithms

Algorithm	ML Technique	Training mode	Attributes	Table update	Parameters/Model update	Energy prediction
MDR-RL	Reinforcement learning	On-line	Deflection ratio Loss probability Hop count to destination Energy status	Yes	*Learning rate* *Recovery coefficient* *Decay coefficient* Q-value	Yes
NNR	Neural Network	Off-line	Hop count to destination Node energy status Loss probability Node popularity	Yes	No	No
DTR	Decision Tree	Off-line	Hop count to destination Node energy status Loss probability Node popularity	Yes	No	No

packet contains reward information, and can be used to update the routing and deflection tables, i.e., there is no updating information in the ACK packets. In the MDR-RL using no updating scheme, all the packets have no updating information and, thus, nano-nodes cannot update the tables. In addition, we study two additional machine learning routing algorithms (NNR and DTR). In NNR and DTR, a delivery probability at the next-hop selection is computed by a trained machine learning model. The message must be only forwarded from the sender to the neighboring receiver node if the intermediate node has a high enough probability of forwarding it directly or indirectly to the destination node. The following attributes are involved in the training processes: hop count to the destination, node energy status, node loss probability and node popularity.

Table 3 compares the complexity of the above algorithm in some aspects. Comprehensively, MDR-RL is more complicated than NNR and DTR. On the one hand, in the MDR-RL, the updating parameters (*learning rate*, *recovery* and *decay coefficient*) need to be updated during every transmission. However, in NNR and DTR, the trained models cannot be updated during the transmissions. On the other hand, in the MDR-RL, an energy prediction scheme is implemented to help the decision-making process, which also increases the complexity.

In the following section, to study the performance of different algorithms, extensive simulations are conducted with different parameters, including the transmission range, density of nano-node, energy harvesting rate and maximum deflection time.

Furthermore, to investigate the influence of different updating parameters in the learning process, including *learning rate* α, *recovery coefficient* β, and *decay coefficient* γ, more simulations are conducted in Sec. 5.7.

5.1 Target Performance Metrics

In this paper, the energy efficiency, network throughput and latency (in terms of packet delivery ratio, number of delivered packets and packet average hop count) of nanonetworks are regarded as the metrics to investigate the performance of different routing algorithms. The packet delivery ratio is obtained by $\frac{N_{del}}{N_{gen}}$, where N_{del} and N_{gen} are the total delivered packet number and generated packets number in the network, respectively. Given the identical simulation duration and energy harvesting rate, higher packet delivery ratio indicates higher energy efficiency. Moreover, the number of delivered packets N_{del} could reflect the throughput of nanonetworks. Larger N_{del} indicates higher throughput of nanonetwork. The latency can be reflected through the average packet hop count. Since larger average packets hop count means that the packets go through more hops to the destinations, it indicates higher latency of nanonetworks.

5.2 Simulation Platform for Energy Harvesting Nanonetworks

To investigate the performance of MDR-RL with different updating schemes, a simulation platform for energy harvesting nanonetworks is developed based on ns-3 and TeraSim [44], [45]. Ns-3 is an open-source project available for researchers, and is widely used for simulation in network systems. TeraSim is built as an extension for ns-3, which enables the networking community to test THz networking protocols without having to delve into the channel and physical layers. In the developed simulator platform, we consider all the nano-nodes equipped with sensing unit, nano-battery, energy harvesting unit, memory unit, processing unit, and transmission unit. The sensing unit can detect from the environment and initiate packet sending requests when energy is sufficient. The energy harvesting unit can harvest energy from the environment to recharge the nano-battery. The memory unit is used to store the packet, routing and deflection tables, and other basic codes. In the simulations, we consider the memory unit can only store one data packet. The processing unit is used to perform the tables update and control process. The responsibility of the transmission unit is to send and receive packets. In the network layer, we consider a random distribution of nano-nodes, and all the nano-nodes have no route entry initially. Moreover, different routing algorithms are implemented to investigate the performance.

In the MAC layer, a simple ALOHA MAC protocol with ACK and NACK is considered. The transmission process ends when the transmitting nano-node receive an ACK or NACK packet from next-hop nano-node, or exceeds the maximum waiting time. In the physical layer, a common modulation scheme is adopted, named Time Spread On-Off Keying (TS-OOK) [46].

Table 4 gives the parameters used in the simulations. The packet sending request interval of nano-node is not fixed, but happens randomly every 1 to 9 seconds. However, a

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TMC.2020.3006535, IEEE Transactions on Mobile Computing

TABLE 4
Parameters used in the simulations

Parameters	Value
Simulation duration	500 s
Density of nano-nodes	$[20-40]\times10^{-2}$ nodes/mm^2
Packet sending request interval	[1 - 9]s
Packet Time To Live	15 hops
Pulse duration	100 fs
Pulse Inter-arrival Time	10 ps
Transmission range of nano-nodes	[4 - 7] mm
Learning rate α Recovery rate β Decay rate γ	[0.1 - 0.9]
Maximum energy capacity of nano-nodes	60 pJ
Average energy harvesting rate	[2 - 4] pJ
Packet size	130 Bytes
ACK size with reward information	13 Bytes
NACK size	9 Bytes
Update process energy	1/30×Transmission energy
Signal-to-noise SNR	10 dB
Boltzmann constant k	1.38×10^{-23} J/K
Bandwidth B	4×10^{12} Hz
Environment temperature T	300 K
Antenna design frequency f	1.6×10^{12} Hz

nano-node does not send packets until it harvests enough energy. The packet TTL is set to 15 hops due to the high density of nanonetworks. The values of pulse duration and pulse inter-arrival time are set according to [43]. Considering the high path loss in the THz band and limited transmission power, the transmission range that a nano-node can achieve is only of a few millimeters. The energy spent in sending and receiving packets depends on the Signal-to-Noise Ratio (SNR) threshold. The relationship between the received pulse power and SNR can be expressed as:

$$\frac{P_{rx}}{N} \geq SNR, \tag{28}$$

where P_{rx} is the received THz signal powers, and N is the noise power, which can be obtained by

$$N = k \cdot B \cdot T, \tag{29}$$

where k refers to the Boltzmann constant, B refers to the bandwidth, and T stands for the environment temperature. According to our earlier work [15], [47], the minimum pulse energy of TS-OOK modulation scheme can be expressed as

$$E_p = P_{tx}T_p = \frac{SNR \cdot k \cdot B \cdot T 16\pi^2 r^2 f^2}{c^2} T_p. \tag{30}$$

where c is the speed of light, r is the distance between transmitter and receiver, f is design frequency of antenna, T_p is pulse duration of TS-OOK modulation scheme.

Furthermore, to study the influences of different parameters in the learning process, the values of *learning rate*, *recovery* and *decay coefficients* are set to range from 0.1 to 0.9.

5.3 Simulations with Different Transmission Ranges

In Figs. 5, 6 and 7, the simulations with different transmission ranges are presented. Here, the nano-nodes density is

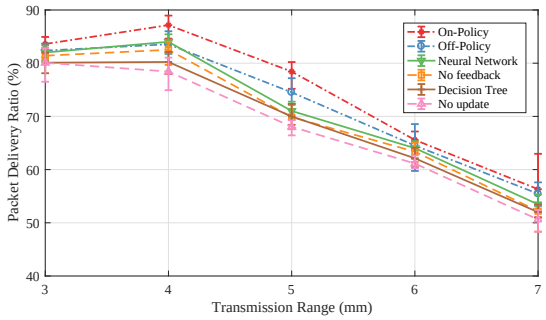


Fig. 5. Packet delivery ratio with different transmission ranges

25×10^{-2} nodes/mm^2, the energy harvesting rate is 3.0 pJ/s, and the maximum deflection time is 1.

Comprehensively, the performance of MDR-RL is better than DTR. The main reason is that, in DTR, the next-hop selection is decided by the off-line trained model, and the parameters of the trained model cannot adapt to the change of network traffic. However, the performance of NNR is better than DTR and MDR-RL with no feedback update policy, but worse than MDR-RL with off-policy and on-policy schemes. On the one hand, the model of NN is more efficient than DT [26], [27]. On the other hand, although the updating and energy prediction processes of MDR-RL need more computational energy than NNR and DTR, it helps the next-hop nano-node selection processes, which improves the network performance.

It also can be observed from Fig. 5 that: (i) the packet delivery ratio when the transmission range is 4 mm is greater than that in 3 mm case. When nano-nodes have a greater transmission range, it is more likely for them to find an adequate routing path. However, according to (30), a higher transmission range also requires a higher transmission power, which leads to greater energy consumption. Hence, when the transmission range exceeds 4 mm, the packet delivery ratio decreases with the increase of transmission range. Therefore, the optimal transmission range needs to be tested when designing the nanonetwork. (ii) The packet delivery ratio of MDR-RL with on-policy updating scheme is greater than it with off-policy updating scheme. In the on-policy updating scheme, a nano-node sends feedback reward of the corresponding routing path which forwards the packet successfully. However, in the off-policy updating scheme, a nano-node sends the feedback reward of the corresponding routing path with minimum Q-value, no matter it forwards the packet successfully or not. This indicates that on-policy updating scheme always chooses the "safest" path to send packets, while the off-policy updating scheme prefers to choose the "shortest" paths. (iii) The packet delivery ratio of MDR-RL with no feedback updating scheme is smaller than on-policy and off-policy updating schemes, but greater than no updating scheme. Although additional updating processes increases the computational energy consumption, it updates the routing and deflection tables to improve the probability to find the more appropriate routing paths to destinations, which increases the packet delivery ratio.

From Fig. 6, it can be concluded that the delivered packet numbers of all routing algorithms decrease with

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TMC.2020.3006535, IEEE Transactions on Mobile Computing

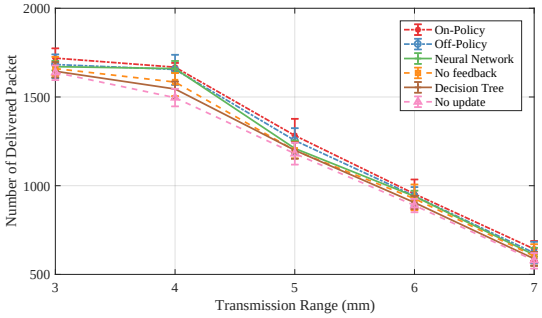


Fig. 6. Number of delivered rackets with different transmission ranges

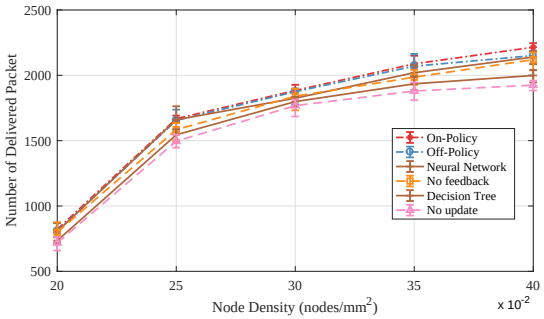


Fig. 9. Number of delivered packets with different nano-node densities

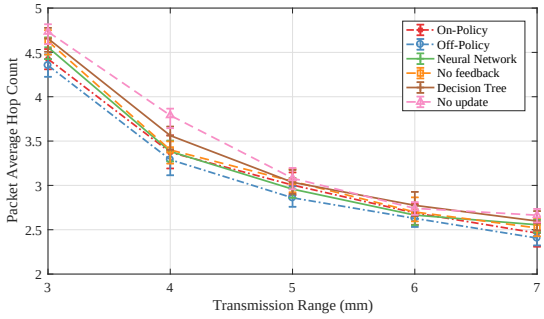


Fig. 7. Packet average hop count with different transmission ranges

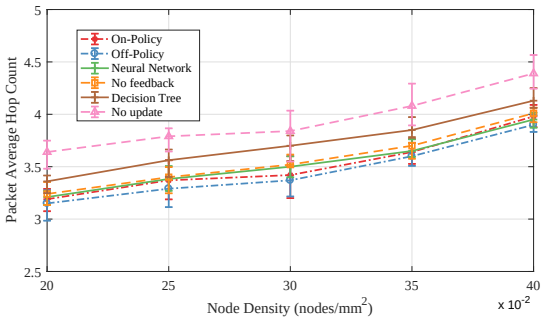


Fig. 10. Packet average hop count with different nano-node densities

transmission range. More energy is required to transmit signals further due to higher path loss.

In Fig. 7, the effects of different transmission ranges on the average hop counts are investigated. Increasing the transmission range helps the nano-node find shorter paths to deliver packets, which decreases the average hop count. The average hop counts of MDR-RL using off-policy updating scheme are smaller than it with on-policy updating scheme, because off-policy updating scheme always tries to choose the "shortest" routing path to update the corresponding routing and deflection tables.

5.4 Simulations with Different Nano-node Densities

The effects of different nano-node densities on the performance are presented in Figs. 8, 9 and 10. In these simulations, the transmission range is set to 4 mm, the energy harvesting rate is 3.0 pJ/s and the maximum deflection time is 1.

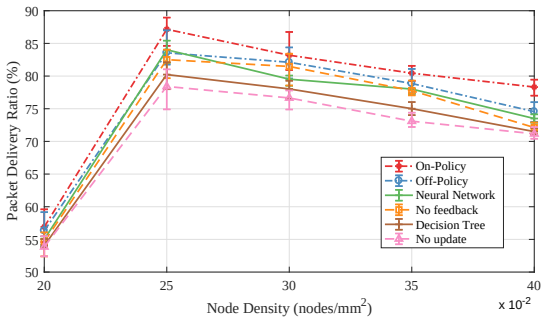


Fig. 8. Packet delivery ratio with different nano-node densities

As shown in Fig. 8, with nano-node densities ranging from 20×10^{-2} to 25×10^{-2} nodes/mm^2, the packet delivery ratios of all routing algorithms increase. On the one hand, the relative distances between nano-nodes decrease with the increase of nano-node density, which makes it easier to find shorter routing paths to destinations. On the other hand, for NNR and DTR, the routing and deflection tables are updated more frequently with more nano-nodes. For MDR-RL, the updating parameters of MDR-RL also can be updated more frequently, which helps nano-nodes converge to better status. However, from 25×10^{-2} to 40×10^{-2} nodes/mm^2 of nano-node density, the packet delivery ratios of all routing algorithm decrease. The probability of finding the optimal routing paths to destination nano-nodes decreases with the increase of nano-node quantity.

Figure 9 illustrates the impact of the nano-node density on the number of delivered packets. We can conclude that the delivered packet numbers of all routing algorithms increase with the density of nano-nodes. The reason is that more packets are generated by increased nano-nodes. However, the MDR-RL can adapt to the change of network traffic load, and can update routing tables and parameters by forwarded data and feedback packets. Therefore, the delivered packet number of MDR-RL can still maintain a higher growth rate.

Finally, as observed in Fig. 10, an increase in the nano-node density increases the packet average hop count of all the routing algorithms. This is because more nano-nodes could increase the hops from sources to destinations. The MDR-RL with off-policy has the lowest packet average hop count.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TMC.2020.3006535, IEEE Transactions on Mobile Computing

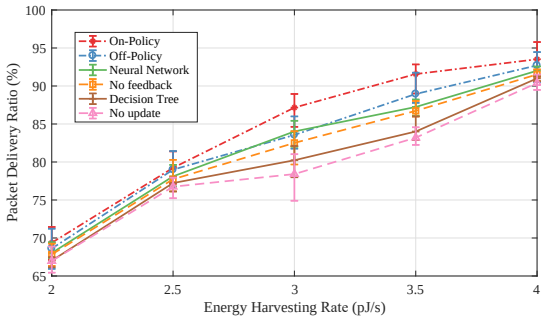


Fig. 11. Packet delivery ratio with different energy harvesting rates

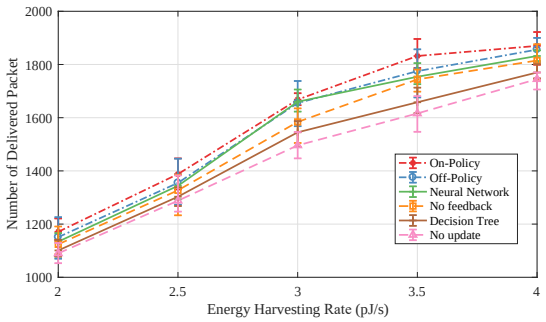


Fig. 12. Number of delivered packets with different energy harvesting rates

5.5 Simulations with Different Energy Harvesting Rates

The effects of different energy harvesting rates on the performance are presented in Figs. 11, 12 and 13. In these simulations, the transmission range is 4 mm, the nano-node density is 25×10^{-2} nodes/mm^2 and the maximum deflection time is 1.

As shown in Fig. 13, the increase of the energy harvesting rate leads to higher average hop count, which indicates higher average network latency. This does not mean that a higher energy harvesting rate results in worse performance. Instead, nano-node can harvest more energy with a higher energy harvesting rate. As a result, more paths become available, ultimately increasing the packet delivery ratio. This can also be verified in Fig. 11. Moreover, as shown in Fig. 12, with a higher energy harvesting rate, nano-nodes can generate more packets.

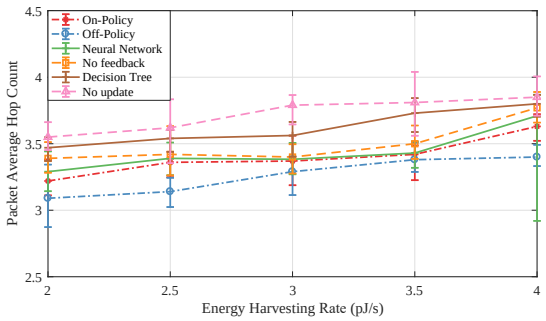


Fig. 13. Packet average hop count with different energy harvesting rates

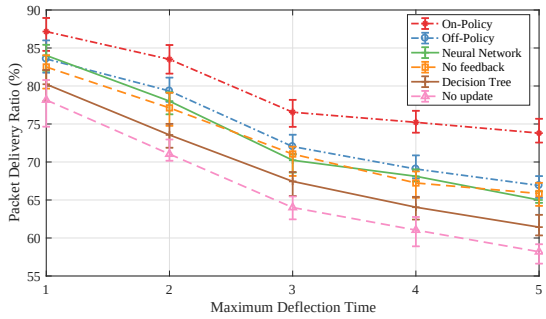


Fig. 14. Packet delivery ratio with different maximum deflection times

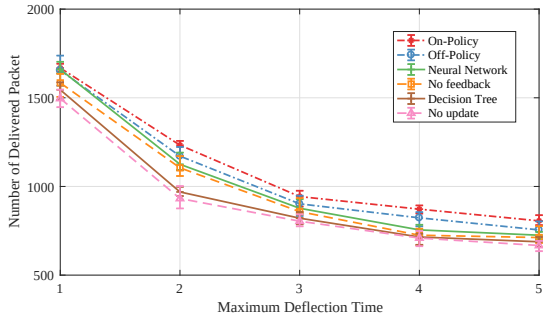


Fig. 15. Number of delivered packets with different maximum deflection times

5.6 Simulations with Different Maximum Deflection Times

In Figs. 14, 15 and 16, the effects of different maximum deflection times on the network performance are presented. In these simulations, the transmission range is set to 4 mm, the energy harvesting rate is 3.0 pJ/s and the nano-node density is 25×10^{-2} nodes/mm^2.

When the route entry in the routing table is invalid, the nano-node deflects the packet to a neighbor decided by the deflection scheme proposed above. After that, if the nano-node still does not receive the ACK packet from the next-hop and deflection time does not exceed maximum deflection time, it deflects the packet to another neighbor.

Theoretically, increasing the deflection time could increase the probability of transmitting packets successfully without considering the energy. However, due to the extremely limited energy capacity, nano-nodes could run out of energy more quickly because of extra deflection. Hence,

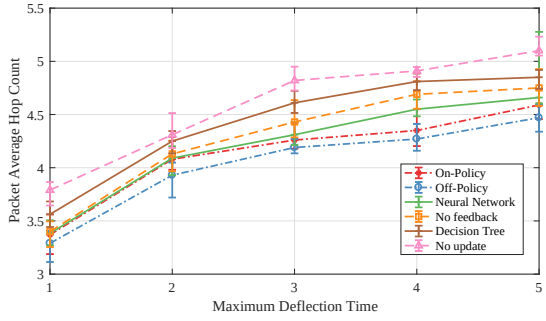
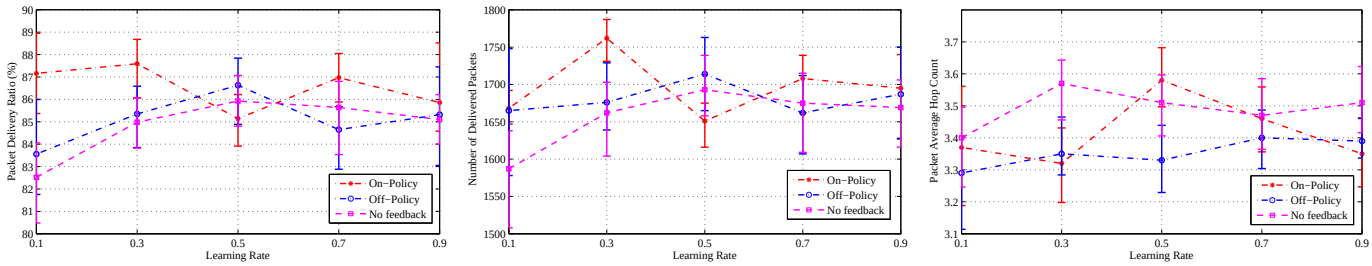


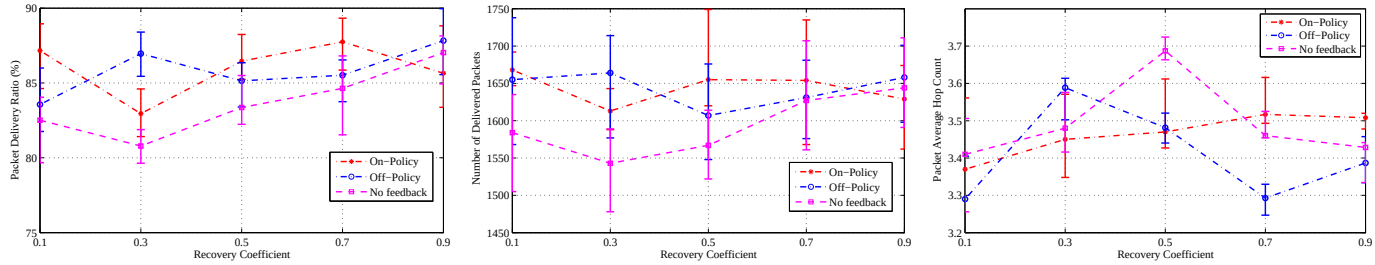
Fig. 16. Packet average hop count with different maximum deflection times

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TMC.2020.3006535, IEEE Transactions on Mobile Computing



(a) Packet delivery ratio (b) Number of delivered rackets (c) Packet average hop count

Fig. 17. Simulations with different *learning rate*



(a) Packet delivery ratio (b) Number of delivered packets (c) Packet average hop count

Fig. 18. Simulations with different *recovery coefficient*

the performance deteriorates with the increase of maximum deflection time. The simulations also verify this: the packet delivery ratio and number of delivered packets decrease with the increase of maximum deflection time, which can be observed from Figs. 14 and 15. In Fig. 16, it can be observed that the average hop count increases with the increase of maximum deflection time, which indicates higher latency of the network.

5.7 Simulations with Different Updating Parameters

To study the influences of the different learning and updating parameters of MDR-RL, including *learning rate*, *recovery* and *decay coefficients*, extensive simulations with different values of these parameters are conducted. In these simulations, the transmission range is 4 mm, the energy harvesting rate is 3.0 pJ/s, the nano-node density is 25×10^{-2} nodes/mm^2 and the maximum deflection time is 1. We only compare the MDR-RL using on-policy, off-policy, and no feedback updating schemes, owing to only these three algorithms have the updating process.

5.7.1 Simulations with Different Learning Rates

In the simulations with different *learning rates*, the *recovery* and *decay coefficients* are set to 0.1 and 0.9, respectively.

It can be concluded from Fig. 17(a) to 17(c) that: (i) comprehensively, the MDR-RL using no feedback updating scheme has the worst performance, since only forwarded packets are utilized to update the routing and deflection tables; (ii) overall, the on-policy updating scheme has better performance on the packet delivery ratio and number of delivered packets, but worse performance on average hop count than off-policy updating scheme. The reason is that on-policy updating scheme always chooses a "safe" routing path to update the routing and deflection tables, but off-policy updating process prefers a "short" one; (iii) in this case, 0.3, 0.5, 0.5 are the best *learning rate* for on-policy, off-policy, and no feedback updating schemes, respectively.

Combining Figs. 17(a) and 17(b), the changing trends of packet delivery ratio and number of delivered packets are consistent with the change of *learning rate*. However, under the condition of consistent energy harvesting rate, higher packet average hop count indicates more energy is spent on forwarding packets, which could increase the transmission failure probability. Therefore, the changing trends of packet delivery ratio and average hop count are inverse, which can be verified in Figs. 17(a) and 17(c).

5.7.2 Simulations with Different Recovery Coefficients

In the simulations with different *recovery coefficients*, the *learning rate* and *decay coefficient* are set to 0.1 and 0.9 respectively.

As shown in Fig. 18(a) to 18(c), comprehensively, in this case, 0.1, 0.9, 0.9 are the best *recovery coefficient* for on-policy, off-policy and no feedback updating schemes, respectively.

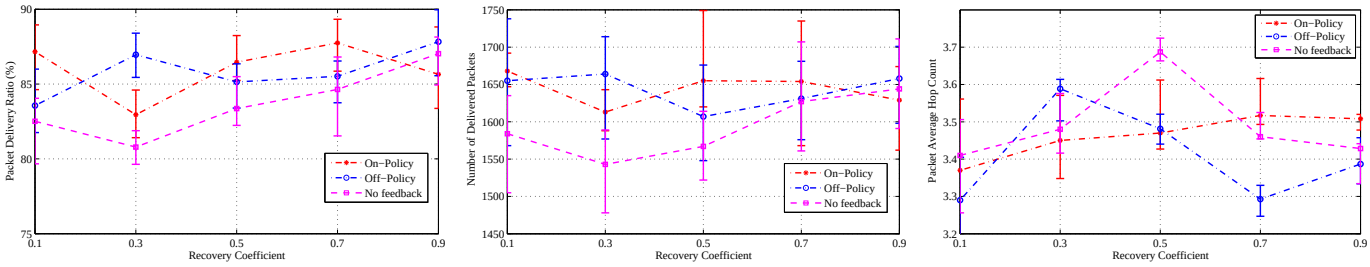
5.7.3 Simulations with Different Decay Coefficients

In the simulations with different *decay coefficients*, the *learning rate* and *recovery coefficient* are both set to 0.1.

It can be observed from Fig. 19(a) to 19(c), 0.9, 0.7, 0.1 are the best *decay coefficient* for on-policy, off-policy and no feedback updating schemes, respectively.

6 Conclusions

A multi-hop deflection routing algorithm based on RL is proposed in this paper to improve the performance of multi-hop communication in nanonetworks. In the algorithm, a



(a) Packet delivery ratio (b) Number of delivered packets (c) Packet average hop count

Fig. 19. Simulations with different *decay coefficient*

new deflection table is established to enable nano-nodes to deflect packets when the next-hop nano-node in the routing table is invalid. Moreover, an energy prediction scheme is implemented to help the deflection decision-making process. The reward information, including Q-value and hop count of the corresponding routing path, packet drop ratio, packet deflect ratio, and energy status of nano-node are considered to update the deflection and routing tables. One forward updating scheme and two feedback updating schemes are proposed to make the nano-node adapt to the dynamic of network (change of status of nano-node and network traffic load) and explore the optimal routing paths to destinations.

Based on extensive simulation results in ns-3, the MDR-RL using on-policy updating scheme has the best performance in terms of packet delivery ratio and number of delivered packets when compared to DTR, NNR, and MDR-RL using off-policy, no feedback, and no updating schemes. In the three kinds of updating schemes, the off-policy updating scheme has the smallest packet average hop count, which indicates the smallest delay. Moreover, the effects of different learning and updating parameters of MDR-RL are presented, the optimal values of learning and updating parameters need to be tested in different scenarios.

Acknowledgments

This work is supported by the National Natural Science Foundation of China (NSFC) under Grant No. 61772471, 61873240.

References

[1] I. F. Akyildiz, M. Pierobon, and S. Balasubramaniam, "Moving forward with molecular communication: from theory to human health applications," *Proceedings of the IEEE*, vol. 107, no. 5, pp. 858–865, 2019.

[2] M. Sharifi, M. R. Avadi, F. Attar, F. Dashtestani, H. Ghorchian, S. M. Rezayat, A. A. Saboury, and M. Falahati, "Cancer diagnosis using nanomaterials based electrochemical nanobiosensors," *Biosensors and Bioelectronics*, vol. 126, pp. 773–784, 2019.

[3] I. F. Akyildiz, J. M. Jornet, and M. Pierobon, "Nanonetworks: A new frontier in communications," *Communications of the Acm*, vol. 54, no. 54, pp. 84–89, 2011.

[4] M. M. U. Rahman, Q. Abbasi, N. Chopra, K. Qaraqe, and A. Alomainy, "Physical layer authentication in nano networks at terahertz frequencies for biomedical applications," *IEEE Access*, vol. 5, no. 99, pp. 7808–7815, 2017.

[5] J. L. Marzo, J. M. Jornet, and M. Pierobon, "Nanonetworks in biomedical applications," *Current drug targets*, vol. 20, no. 8, pp. 800–807, 2019.

[6] A. Afsharinejad, A. Davy, B. Jennings, and C. Brennan, "Performance analysis of plant monitoring nanosensor networks at thz frequencies," *IEEE Internet of Things Journal*, vol. 3, no. 1, pp. 59–69, 2016.

[7] N. Dasgupta, S. Ranjan, and C. Ramalingam, "Applications of nanotechnology in agriculture and water quality management," *Environmental Chemistry Letters*, vol. 15, no. 4, pp. 591–605, 2017.

[8] G. Fuertes, I. Soto, R. Carrasco, M. Vargas, J. Sabattin, and C. Lagos, "Intelligent packaging systems: sensors and nanosensors to monitor food quality and safety," *Journal of Sensors*, vol. 2016, 2016.

[9] A. Nayyar, V. Puri, and D.-N. Le, "Internet of nano things (iont): Next evolutionary step in nanotechnology," *Nanoscience and Nanotechnology*, vol. 7, no. 1, pp. 4–8, 2017.

[10] J. M. Jornet and I. F. Akyildiz, "Graphene-based plasmonic nano-antenna for terahertz band communication in nanonetworks," *IEEE Journal on Selected Areas in Communications*, vol. 31, no. 12, pp. 685–694, 2013.

[11] R. Pan, R. Sun, Z. Wang, J. Lindh, K. Edström, M. Strømme, and L. Nyholm, "Sandwich-structured nano/micro fiber-based separators for lithium metal batteries," *Nano Energy*, vol. 55, pp. 316–326, 2019.

[12] G. Hills, C. Lau, A. Wright, S. Fuller, M. D. Bishop, T. Srimani, P. Kanhaiya, R. Ho, A. Amer, Y. Stein *et al.*, "Modern microprocessor built from complementary carbon nanotube transistors," *Nature*, vol. 572, no. 7771, pp. 595–602, 2019.

[13] C.-C. Chang, C.-C. Lu, M.-C. Wang, H.-S. Huang, S.-Y. Chen, and S.-J. Wang, "Nano-node n-type gate dielectric integrity and uniformity correlated to nitridation process," in *2019 8th International Symposium on Next Generation Electronics (ISNE)*. IEEE, 2019, pp. 1–3.

[14] S. Anand, D. S. Kumar, R. J. Wu, and M. Chavali, "Graphene nanoribbon based terahertz antenna on polyimide substrate," *Optik - International Journal for Light and Electron Optics*, vol. 125, no. 19, pp. 5546–5549, 2014.

[15] X.-W. Yao, W.-L. Wang, and S.-H. Yang, "Joint parameter optimization for perpetual nanonetworks and maximum network capacity," *IEEE Transactions on Molecular, Biological and Multi-Scale Communications*, vol. 1, no. 4, pp. 321–330, 2015.

[16] J. Kokkoniemi, J. Lehtomaki, K. Umebayashi, and M. Juntti, "Frequency and time domain channel models for nanonetworks in terahertz band," *IEEE Transactions on Antennas and Propagation*, vol. 63, no. 2, pp. 678–691, 2015.

[17] M. Zainuddin, E. Dedu, and J. Bourgeois, "Low weight code comparison for electromagnetic wireless nanocommunication," *IEEE Internet of Things Journal*, vol. 3, no. 1, pp. 38–48, 2017.

[18] X.-W. Yao, D.-B. Ma, and C. Han, "Ecp: A probling-based error control strategy for thz-based nanonetworks with energy harvesting," *IEEE Access*, 2019, In press.

[19] W.-L. Wang, C.-C. Wang, and X.-W. Yao, "Slot self-allocation based mac protocol for energy harvesting nano-networks," *Sensors*, vol. 19, no. 21, p. 4646, 2019.

[20] X.-W. Yao, C.-C. Wang, W.-L. Wang, and J. M. Jornet, "On the achievable throughput of energy-harvesting nanonetworks in the terahertz band," *IEEE Sensors Journal*, vol. 18, no. 2, pp. 902–912, 2018.

[21] J. M. Jornet and I. F. Akyildiz, "Channel modeling and capacity analysis for electromagnetic wireless nanonetworks in the terahertz band," *IEEE Transactions on Wireless Communications*, vol. 10, no. 10, pp. 3211–3221, 2011.

[22] C. Liu, E. I. Gillette, X. Chen, A. J. Pearse, and G. W. Rubloff, "An all-in-one nanopore battery array," *Nature Nanotechnology*, vol. 9, no. 12, 2014.

[23] L. P. Zaino, C. Ma, and P. W. Bohn, "Nanopore-enabled electrode arrays and ensembles," *Mikrochimica Acta*, vol. 183, no. 3, pp. 1019–1032, 2016.
[24] W. Liu, Z. Wang, G. Wang, G. Liu, J. Chen, X. Pu, Y. Xi, X. Wang, H. Guo, C. Hu *et al.*, "Integrated charge excitation triboelectric nanogenerator," *Nature communications*, vol. 10, no. 1, pp. 1–9, 2019.
[25] L. Gu, J. Liu, N. Cui, Q. Xu, T. Du, L. Zhang, Z. Wang, C. Long, and Y. Qin, "Enhancing the current density of a piezoelectric nanogenerator using a three-dimensional intercalation electrode," *Nature Communications*, vol. 11, no. 1, pp. 1–9, 2020.
[26] A. Mehmood, Z. Lv, J. Lloret, and M. M. Umar, "Eldc: An artificial neural network based energy-efficient and robust routing scheme for pollution monitoring in wsns," *IEEE Transactions on Emerging Topics in Computing*, 2017.
[27] X. Li, B. Keegan, F. Mtenzi, T. Weise, and M. Tan, "Energy-efficient load balancing ant based routing algorithm for wireless sensor networks," *IEEE Access*, vol. 7, pp. 113 182–113 196, 2019.
[28] N. Abuali, S. Aleyadeh, F. Djebbar, A. Alomainy, M. M. A. Almaazmi, and S. A. Ghaithi, "Performance evaluation of routing protocols in electromagnetic nanonetworks," *IEEE Access*, vol. 6, pp. 35 908–35 914, 2018.
[29] A. Tsioliaridou, C. Liaskos, S. Ioannidis, and A. Pitsillides, "Corona: A coordinate and routing system for nanonetworks," in *International Conference on Nanoscale Computing and Communication*, 2015, pp. 1–6.
[30] C. Liaskos, A. Tsioliaridou, S. Ioannidis, N. Kantartzis, and A. Pitsillides, "A deployable routing system for nanonetworks," in *2016 IEEE International Conference on Communications (ICC)*. IEEE, 2016, pp. 1–6.
[31] T. Ageliki, C. K. Liaskos, E. Dedu, and S. Ioannidis, "Stateless linear-path routing for 3d nanonetworks," in *ACM International Conference on Nanoscale Computing and Communication*, 2016, pp. 1–6.
[32] Y. Chen, C. Qiao, and X. Yu, "Optical burst switching: a new area in optical networking research," *IEEE Network Magazine*, vol. 18, no. 3, pp. 16–23, 2004.
[33] N. Akar, E. Karasan, K. G. Vlachos, E. A. Varvarigos, D. Careglio, M. Klinkowski, and J. Sole-Pareta, "A survey of quality of service differentiation mechanisms for optical burst switching networks," *Optical Switching and Networking*, vol. 7, no. 1, pp. 1–11, 2010.
[34] S. S. Chawathe, "Analysis of burst header packets in optical burst switching networks," in *IEEE International Symposium on Network Computing and Applications*, 2018.
[35] P. Khumalo, B. Nleya, and A. Mutsvangwa, "A controllable deflection routing and wavelength assignment algorithm in obs networks," *Journal of Optics*, vol. 48, no. 4, pp. 539–548, 2019.
[36] M. Thachayani and R. Nakkeeran, "Combined probabilistic deflection and retransmission scheme for loss minimization in obs networks," *Optical Switching and Networking*, vol. 18, pp. 51–58, 2015.
[37] Y. Li, K. Mei, Y. Liu, N. Zheng, and Y. Xu, "Ldbr: Low-deflection bufferless router for cost-sensitive network-on-chip design," *Microprocessors and Microsystems*, vol. 38, no. 7, pp. 669–680, 2014.
[38] A. Belbekkouche, A. Hafid, and M. Gendreau, "Novel reinforcement learning-based approaches to reduce loss probability in buffer-less obs networks," *Computer Networks*, vol. 53, no. 12, pp. 2091–2105, 2009.
[39] S. Chettibi and S. Chikhi, "Dynamic fuzzy logic and reinforcement learning for adaptive energy efficient routing in mobile ad-hoc networks," *Applied Soft Computing*, vol. 38, pp. 321–328, 2016.
[40] C.-C. Wang, Q. Xia, X.-W. Yao, W.-L. Wang, and J. M. Jornet, "Multi-hop deflection routing algorithm based on Q-Learning for energy-harvesting nanonetworks," in *The 15th International Conference on Mobile Ad-hoc and Sensor Systems (IEEE MASS 2018)*, Chengdu, P.R. China, Oct. 2018.
[41] C. J. C. H. Watkins and P. Dayan, "Q-learning," in *Machine Learning*, 1992, pp. 279–292.
[42] C. Szepesvári, "The asymptotic convergence-rate of q-learning," in *Advances in Neural Information Processing Systems*, 1998, pp. 1064–1070.
[43] J. M. Jornet, J. C. Pujol, and J. S. Pareta, "Phlame: A physical layer aware mac protocol for electromagnetic nanonetworks in the terahertz band," *Nano Communication Networks*, vol. 3, no. 1, pp. 74–81, 2012.
[44] G. F. Riley and T. R. Henderson, "The ns-3 network simulator," in *Modeling and tools for network simulation*. Springer, 2010, pp. 15–34.
[45] J. M. Jornet, "Terasim: An ns-3 extension to simulate terahertz-band communication networks," *Nano communication networks*, vol. 17, no. SEP., pp. 36–44.
[46] J. M. Jornet and I. F. Akyildiz, "Femtosecond-long pulse-based modulation for terahertz band communication in nanonetworks," *IEEE Transactions on Communications*, vol. 62, no. 5, pp. 1742–1754, 2014.
[47] X.-W. Yao, C.-C. Wang, W.-L. Wang, and C. Han, "Stochastic geometry analysis of interference and coverage in terahertz networks," *Nano communication networks*, vol. 13, pp. 9–19, 2017.

Chao-Chao Wang received the Ph.D. degree in control science and engineering from the Zhejiang University of Technology, Hangzhou, China, in 2020. Currently, he is a lecturer with the College of Mathematics, Physics and Information Engineering at the Jiaxing University, Jiaxing, China. From December 2017 to June 2016, he was a visiting student at the University of Buffalo, The State University of New York. His current research interests include Terahertz band communication networks, Electromagnetic Nanonetwork, Wireless Ad Hoc Sensor networks, Internet of Things, 5G Cellular Networks.

Xin-Wei Yao received the Ph.D. degree in information engineering from the Zhejiang University of Technology, Hangzhou, China, in 2013. He is currently an associate professor with the College of Computer Science and Technology at the Zhejiang University of Technology, Hangzhou, China. From March 2012 to February 2013, he was a visiting scholar at the Loughborough University, Leicestershire, UK. From August 2015 to July 2016, he was a visiting professor at the University of Buffalo, The State University of New York, Buffalo, NY, USA. He was the recipient of the Distinguished Associate Professor Award and the Outstanding Doctoral Thesis Award at the Zhejiang University of Technology. He has served on technical program committees of many IEEE/ACM conferences. He is a member of the IEEE and the ACM. His current research interests are in the area of Terahertz-Band Communication Networks, Electromagnetic Nanonetworks, Wireless Ad Hoc and Sensor networks, Wireless Power Transfer and the Internet of Things.

Wan-Liang Wang received his Ph.D. degree from TongJi University, Shanghai, China, in 2001. Now, he is a professor in Zhejiang University of Technology, Hangzhou, China. His research interests include intelligent algorithms and network control. He has published over 200 refereed papers.

Josep Miquel Jornet (M'13) received the B.S. in Telecommunication Engineering and the M.Sc. in Information and Communication Technologies from the Universitat Politècnica de Catalunya, Barcelona, Spain, in 2008. He received the Ph.D. degree in Electrical and Computer Engineering from the Georgia Institute of Technology (Georgia Tech), Atlanta, GA, in 2013. From September 2007 to December 2008, he was a visiting researcher at the Massachusetts Institute of Technology (MIT), Cambridge, under the MIT Sea Grant program. Between August 2013 and August 2019, he was a faculty with the Department of Electrical Engineering at the University at Buffalo, The State University of New York. Since August 2019, he is an Associate Professor in the Department of Electrical and Computer Engineering, the Director of the Ultrabroadband Nanonetworking Laboratory, and a member of the Institute for the Wireless Internet of Things at Northeastern University, in Boston, MA. His current research interests are in Terahertz-band communication networks, Wireless Nano-bio-communication Networks and the Internet of Nano-Things. In these areas, he has co-authored more than 140 peer-reviewed scientific publications, 1 book, and has also been granted 3 US patents. Since July 2016, he is the Editor-in-Chief of the Nano Communication Networks (Elsevier) Journal. He is serving as the lead PI on multiple grants from U.S. federal agencies including the National Science Foundation, the Air Force Office of Scientific Research and the Air Force Research Laboratory. He is a recipient of the National Science Foundation CAREER award and of several other awards from IEEE, ACM and UB.