

A lightweight forwarding strategy for Named Data Networking in low-end IoT

Amar Abane, Mehammed Daoui, Samia Bouzefrane, Paul Mühlethaler

▶ **To cite this version:**

Amar Abane, Mehammed Daoui, Samia Bouzefrane, Paul Mühlethaler. A lightweight forwarding strategy for Named Data Networking in low-end IoT. Journal of Network and Computer Applications (JNCA), 2019, 148, pp.102445. 10.1016/j.jnca.2019.102445 . hal-02334088

HAL Id: hal-02334088
https://hal.science/hal-02334088

Submitted on 25 Oct 2019

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

A lightweight forwarding strategy for Named Data Networking in low-end IoT

Amar Abane[a,b], Mehammed Daoui[a], Samia Bouzefrane[b,*], Paul Muhlethaler[c]

[a] *University Mouloud Mammeri of Tizi-Ouzou (Algeria)*
[b] *Conservatoire National des Arts et Metiers, Paris (France)*
[c] *Inria, Paris (France)*

Abstract

The proliferation of IoT devices has given a new dimension to networking operations. In fact, IoT issues in terms of complex communication patterns, dynamic topologies and security force to re-think the basic networking operations in a more adapted way. Moreover, the IoT needs contribute to show the limitations of the IP model to support content-based applications, and the emergence of the Information-Centric Networking (ICN) paradigm seems to be more compliant with the IoT vision. In this context, to enable the ICN model in IoT devices, we present a lightweight forwarding strategy for Named Data Networking over IEEE802.15.4. Named Data Networking is an ICN architecture with a great potential for the IoT and future Internet. This paper presents a forwarding strategy that reduces network overhead to the bare minimum, while keeping satisfactory performance in different IoT application scenarios. To forward named contents without node addresses, the strategy is based on a reinforcement learning technique that provides an accurate broadcast-based forwarding with a reduced overhead.

Keywords: Named Data Networking, learning, wireless networks, Internet of Things, IEEE802.15.4.

1. Introduction

Massive technological innovations towards electronic miniaturization, associated with the deployment of manifold wireless technologies, have fostered the emergence of billions of small devices connected over the Internet, creating the omnipresent *Internet of Things* (IoT) [1]. IoT systems are currently deployed on the IP protocol suite. However, the IP architecture shows its limitations to support the IoT needs. For example, security is still focused on communication channels when the data itself needs to be secured. Moreover, IoT systems need

*Corresponding author
Email address: samia.bouzefrane@cnam.fr (Samia Bouzefrane)

efficient support for resource naming and discovery, which is not easy to deploy with IP in constrained infrastructures [2]. Although dedicated adaptations such as CoAP and 6LoWPAN have been developed, matching the IoT vision without fundamental changes remains difficult for IP.

The Information Centric Networking (ICN) [3] approach can natively support IoT needs including resource identification, access control, content security, and mobility. The most interesting embodiment of the ICN model is Named Data Networking (NDN) [4]. In NDN, pieces of contents are named independently of their location, following a URL-like naming structure. Applications request contents from the network by sending *Interest* packets that pull back *Data* packets. Native features come along with this principle such as: communication without establishing end-to-end connections and a name-to-address resolution. In addition, it is not necessary to maintain consumer-provider paths, which provides a native support of connection disruption resulting from node mobility.

So far, the suitability of NDN for the IoT has been investigated to some extent [5], but the lightweight aspect of NDN should be improved in order to enable NDN in current IoT systems. It has been observed in [6] that an NDN stack implementation in IoT devices (RIOT and Contiki) can save up to *60%* of ROM and *80%* of RAM compared to the RPL/6LoWPAN stack. Moreover, a simple flooding mechanism in an NDN wireless network generates three times fewer packets than RPL/6LoWPAN routing. Even if the superiority of NDN in terms of performance remains to be demonstrated, the data-centric approach of NDN can be an alternative to the 6LoWAPN stack in future IoT solutions [7].

To deploy NDN in IoT devices, one of the first challenges is to support the NDN forwarding in a low-rate wireless network, such as IEEE802.15.4. NDN wireless forwarding strategies are generally based on a *broadcast-and-learn* mechanism. For example, such an approach, which can be called Blind Flooding (BF)[8] has been proposed for NDN over IEEE802.11. Nodes use timers to defer Interest and Data packet retransmissions, so that a node can cancel its retransmission if another node is sending an Interest/Data with the same name. However, such a solution cannot be used directly over IEEE802.15.4 due to data-rate and resource limitations.

Although lightweight mechanisms for constrained devices are rare in the literature, a simple one has been proposed in [6]. This technique called RONR (detailed in Section 3) uses Interest broadcast (flooding) to discover the requested content. After the Data packet is received, each node keeps a mapping between the name prefix of the content and the source MAC address of the received Data, which will be used as the next hop address to forward subsequent Interests. The flooding phase is triggered again when the mapping expires. However, we show in this paper that unicast is not always the best approach, especially with mobile devices, due to the reusable nature of NDN packets as they are independent from host addresses.

To take advantage of the broadcast while reducing overhead, we design a Reinforcement-based Lightweight Forwarding (R-LF) strategy for NDN over IEEE802.15.4. The idea of R-LF is to make the broadcast as accurate as the uni-

cast, in terms of transmitted frames and retransmission decisions. Instead of targeting optimal performance for one configuration, which can be achieved more efficiently with a proactive mechanism, our objective is to design a lightweight *tradeoff* technique that can keep satisfactory performance in different communication patterns such as multiple data flows, multi-consumer, multi-source, and dynamic topology changes.

Our contribution through this paper can be summarized as follows:

- We study the importance of the broadcast mechanism and we draw up guidelines for designing a forwarding strategy over IEEE802.15.4.
- We propose the R-LF strategy based only on content names and broadcast communications.
- We show by simulation that our strategy can fit different small-scale IoT scenarios in comparison to all-unicast and all-broadcast forwarding approaches.
- We deploy NDN with R-LF on Arduino devices and we give measures on its lightness aspect.

The rest of this paper is organized as follows: Section 2 gives an overview of NDN, focusing on the forwarding aspect. Section 3 presents relevant forwarding approaches for NDN in wireless networks. In Section 4, we show the importance of the broadcast in wireless NDN forwarding, followed by our guidelines for a lightweight forwarding strategy design. Section 5 details the proposed strategy and its relevant steps, operations and parameters. Section 6 reports the results of the strategy evaluation in different communication scenarios. Section 7 concludes the paper with a brief discussion and future improvements.

2. Named Data Networking

NDN is an ICN protocol in which the communication is completely based on content names, while the IP protocol uses source and destination addresses to deliver contents.

In NDN, content chunks are decoupled from the producer location, and each data packet must be signed by its producer. Thereby, routers can reuse the same packet to satisfy other consumer requests for the same content, providing NDN with a native caching mechanism. The connectionless communication model of NDN gives it a native mobility support, and relying exclusively on content names to communicate removes the need for middleware to support applications [9].

Consequently, NDN can be seen as an evolved network protocol, since it intrinsically includes operations traditionally provided by higher layers in the IP model, such as security and flow control.

NDN defines two types of packets: Interest and Data, and each NDN node maintains three data structures to ensure its operations: the Forwarding Information Base (FIB) used by a node to decide which interface(s) to forward

the Interest to, the Pending Interest Table (PIT) contains Interest already forwarded waiting for Data, and the Content Store (CS) contains forwarded Data packets cached to satisfy future requests.

NDN contents are identified through hierarchical natural names, formed by a sequence of name components separated by slashes. Each component can be accessed and handled separately. For example, the name */home/room1/temperature* may identify the temperature value of room 1, while */home/room1/humidity* identifies the humidity value in room 1, and */home/room2/all* identifies all sensor values of room 2. Applications are free to design their own naming scheme since the routers do not interpret the whole name. Moreover, a producer can add name components to the initial name in order to provide more information about the content (e.g. timestamp, sequence number, geolocation, etc.).

The communication is consumer driven, that is, a communication is initiated by the consumer who asks for the content. Typically, an NDN exchange operates according to the following steps: (1) The consumer requests a certain content by sending an Interest carrying the name of the requested data (e.g. */room1/temperature*). (2) The router that receives an Interest checks if the corresponding Data exists in the CS; if the Data is found it is returned to satisfy the Interest without going further. Otherwise, the router verifies if an Interest requesting the same content is already in the PIT: if so the Interest is dropped and only the incoming interface is added to the existing PIT entry. If there is no Interest waiting in the PIT, the router checks the FIB to find one or multiple interfaces to forward the Interest (i.e. multipath routing supported) and creates a PIT entry containing the forwarded Interest with its incoming interface. (3) When an Interest reaches the node that contains the requested content, this node sends back a Data packet carrying the content, the name that is the same as the Interest name to which suffixes may be added, and security information about the content producer (e.g. signature, key name, etc.). This Data packet takes the reverse path of the Interest following traces left in the PIT of each intermediate node. Each Data packet is a response to one Interest packet.

Figure 1 summarizes the processing steps of an Interest and a Data packet over the NDN data structures. Notice that a matching in the FIB corresponds to the longest prefix match. To forward an Interest named */home/room2/all* for example, assuming there are in the FIB two prefixes */home* and */home/room2*, both prefixes match but the longest one is chosen.

To enable the NDN communication model in wireless networks, some forwarding approaches have been proposed. The next Section reviews the most relevant ones.

3. Related work

The forwarding approaches described in this Section can be defined as *"flood-and-learn"* mechanisms [10], in the sense that a flooding phase is used to discover content sources (i.e. producers or caches), and subsequent requests are forwarded more accurately based on the *learned* information. In the context of

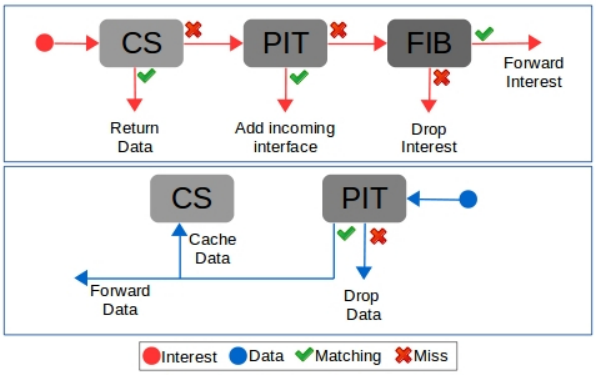


Figure 1: Interest and Data process in NDN

this paper, the flooding phase refers to the sequence of Interest broadcast used by the nodes to retrieve a content when no information is available.

Even if most of these solutions are designed for MANETs and do not consider constrained IoT devices, they represent an interesting starting point to design a wireless NDN forwarding strategy.

In [11], a broadcasting protocol has been designed for data dissemination in vehicular networks. Timers were used to defer packet forwarding. During the waiting time, if a node overhears a packet with the same name, the scheduled packet is dropped, which reduces risks of collisions. However, this blind-flooding mechanism causes a large overhead and does not guarantee that the best path is used to retrieve contents. Therefore, more elaborate solutions have been proposed to move from a blind-flooding to a controlled-flooding.

In the *Listen First Broadcast Later* strategy (LFBL) [12], each node uses its eligibility value to deduce the delay to wait before retransmitting an Interest. The node's eligibility is based on its distance from the content source that each node maintains: the closer is the node to the source, the shorter is the waiting time. This approach contributes to generate reduced overhead by propagating packets faster. However, LFBL nodes maintain distance tables, and endpoint identification (e.g. MAC addresses) is used for source and destination nodes, which slightly evokes the host-based communication model.

The *Neighborhood-Aware Interest Forwarding* (NAIF)[13] strategy is based on a forwarding rate adjustment. Each node collects statistics on its neighborhood forwarding and periodically adjusts its forwarding rate (i.e. the ratio of the Interests it should forward). This mechanism makes the nodes collaboratively forward packets without explicit communication, and no additional data structure is needed. Moreover, good performance results have been obtained in terms of completion ratio, but the round-trip time is quite high. As it is based on a purely statistical method, nodes need to first get enough information and reactively adapt to changes in the situation. This makes the approach slow to converge under dynamical communication changes. Thus, it is more suitable for

downloading contents (e.g. files) but is not adapted for small content chunks of IoT applications. According to the reported results, it achieves almost the same performance as the blind-flooding strategy under high mobility cases.

One forwarding strategy has been implemented in a network of constrained IoT devices over IEEE802.15.4 in [6]. The objective is to reduce the overhead while keeping a satisfactory data retrieval efficiency, all with a minimum state in nodes. This approach called Reactive Optimistic Name-based Routing (RONR) works as follows: after retrieving the first content by Interest broadcast, a node keeps a temporary FIB entry in order to avoid flooding subsequent Interests. The FIB entry binds the content prefix to the content source MAC address. The nodes also use the Interest source MAC address to forward the requested Data packet. This enables a mapping which can be denoted as *Interest Unicast Data Unicast* (IUDU). When the FIB entry for the requested content does not exist, or it is deleted, the flooding phase is performed again to discover another content source. The RONR approach significantly reduces the network overhead. However, the strategy has no means to select the shortest path to retrieve contents. Moreover, we can intuitively figure out that a unicast mapping hardly supports node mobility and simultaneous data flows in a local network.

Table 1 summarizes the mentioned forwarding approaches with their main pros and cons.

Table 1: NDN wireless forwarding approaches

Name	Approach principle	Pros	Cons	Evaluation
BF	Interest flooding Random-delayed retransmissions	Satisfaction rate Mobility support	High overhead Packet redundancy	Simulation (IEEE802.11)
LFBL	Proportional-delay retransmission Content source identification	Low round-trip time Low overhead	Distance table Node identification Mobility	Simulation (IEEE802.11)
NAIF	Adaptive Forwarding rate	Mobility support Satisfaction rate	High round-trip time High overhead	Simulation/Emulation (IEEE802.11)
RONR	Interest flooding Prefix to MAC address mapping	Low round-trip time Low overhead	Mobility NDN potential limited by unicast	Real-world (IEEE802.15.4)

In [14], the name-to-address mapping is studied in terms of resource consumption in low-end IoT devices. Real-world experiments have shown that forwarding packets with unicast addresses can reduce resource consumption in IoT devices and increase the performance in comparison to the broadcast forwarding. However, the studied mappings are not evaluated in terms of network overhead and content dissemination efficiency with mobile nodes. Therefore, in the next Section, we study the impact of NDN-MAC mapping on the communication performance under mobility conditions.

4. NDN-MAC mapping and design guidelines

In this section, we simulate a simple IoT scenario to show the impact of NDN-MAC mapping on forwarding performance. Using the unicast approach described above (RONR), we evaluate other mapping combinations. Therefore, instead of using a unicast address to send both Interest and Data packets, nodes can use different mappings for Interest and Data: (i) Interest Broadcast Data

Broadcast (IBDB), (ii) Interest Broadcast Data Unicast (IBDU), (iii) Interest Unicast Data Broadcast (IUDB), (iv) Interest Unicast Data Unicast (IUDU).

We evaluated these mappings in a network of *16* relay nodes, with *4* mobile consumers requesting contents from *1* mobile producer. A cache of *20* packets is enabled in relay nodes. Details on parameters and metrics are provided in Section 6.

According to the results reported in Figure 2, mappings with Interest broadcast (i.e. IBDB and IBDU) always transmit the highest number of frames as expected. Indeed, the broadcast generates more Interests in the network, which leads to more retransmissions and more collisions. For the same reason, both IUDU and IUDB generate the lowest number of frames.

IBDB and IUDB achieve, respectively, the first and the second best Interest satisfaction rate. This shows an advantage of Data broadcast, and NDN in general, to satisfy multiple pending Interests with one Data packet transmission. Moreover in IBDB, the exploratory nature of Interest broadcast takes advantages of caching in relay nodes, which gives the best satisfaction rate. This is confirmed by the mean hop count of received Data, which is the smallest in IBDB and the highest in IUDU.

However, a significant difference is observed in the Interest-Data round-trip time. Data broadcast in IUDB as well as in IBDB causes a high medium-access competition, which leads to a higher RTT than with Data unicast (i.e. IUDU and IBDU). This raises an important concern about the necessity of using accurate timers to defer packet retransmissions. At the same time, since mappings with Data broadcast provide the highest satisfaction rate, we can not design a forwarding strategy for NDN without considering the Data broadcast.

To sum up, the results suggest to consider Interest and Data broadcast, as it increases content dissemination efficiency and provides a good Interest satisfaction rate. However, the large overhead caused by the Interest broadcast requires a careful design to reduce unnecessary Interest retransmissions. Moreover, the vision of NDN that consists in retrieving contents without using any host address is nicely satisfied with a broadcast strategy.

The simulation results and the study of the related solutions allow to draw up the following guidelines for an adapted forwarding strategy: (i) rely on a minimal state to process packets without maintaining explicit routes, (ii) avoid reverting to a random Interest flooding phase in order to reduce collision risks, network congestion and overhead, (iii) avoid node identification and additional data structures to correctly fit the NDN vision, (iv) distribute decisions and computation tasks over the network, and minimize complexity to meet IoT device capabilities.

Based on these guidelines, we designed an approach inspired from [15] that uses the learned values to compute the delay time to wait before retransmitting a packet. However, the framework has been designed to handle simple sensor-to-sink communications in WSNs, which makes it unsuitable for complexe communication modes. Thus, fundamental improvements and accurate parameter calibration have been made to build an adaptable forwarding strategy for wireless named data networks as detailed in the next Section.

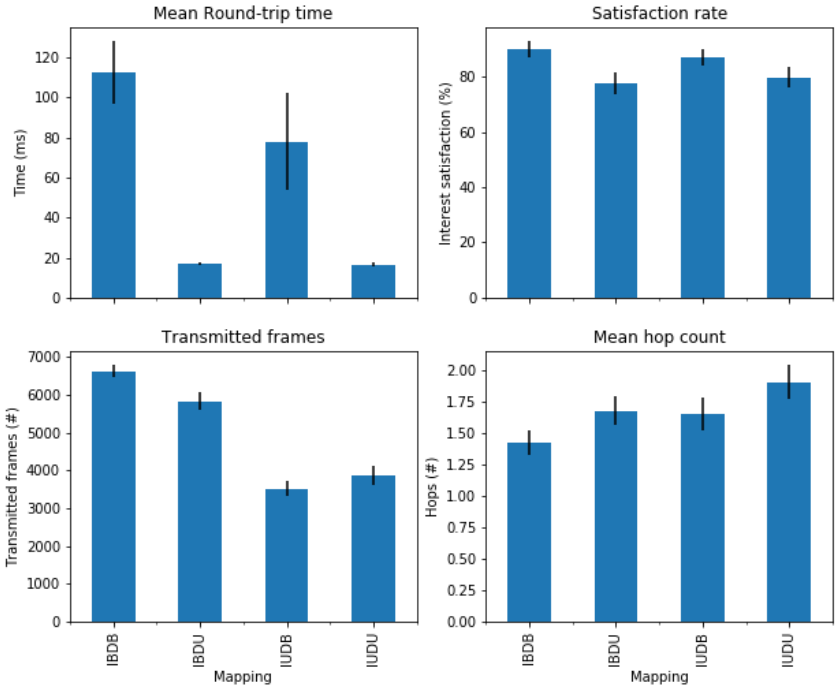


Figure 2: NDN-MAC mapping simulation results

5. R-LF strategy design

5.1. Overview

Routing and forwarding operations are significantly different in NDN and IP. In IP, only the routing operation is smart in the sense that different routing protocols can be envisioned. The forwarding operation always consists in finding the longest match available in the routing table and sending the packet to the corresponding next hop. In NDN however, in addition to the routing operation that can be smart as in IP, multiple approaches are possible to handle packet forwarding with more or less additional information and with or without caching.

Our proposed strategy does not use an explicit routing phase to provide/update forwarding information[1]. R-LF is a *learn-broadcast-learn* approach that operates according to the following steps: (i) the nodes overhear Data packets and learn a cost value by reinforcement, (ii) the nodes decide to forward an Interest

[1]The routing phase is implicit as in the mechanisms presented in the related work.

with a delay according to their cost-based eligibility, (iii) the nodes update their cost from the result, which can be an Interest timeout or a received Data packet.

The following describes the general R-LF approach and the mathematical formalization.

5.2. General description

To forward packets, a node traditionally decides in terms of what the next hop is, which can be considered as a spatial forwarding decision. However, as NDN forwarding is based on content names and R-LF runs directly on top on the MAC layer, a node decides in terms of when it should retransmit the Interest, or how long it should wait before retransmitting. Such a process can be seen as a temporal forwarding decision. As reported above, this vision has already been proposed, and its basic principle consists in ensuring that the more eligible node will retransmit the Interest first.

To describe the forwarding approach, the following assumptions are made:

- Interest and Data packets carry the cost value of the sending node, denoted as *C-field*.
- An Interest flooding with random delays is used to find the producer when the first Interest is issued for an unknown content.
- Nodes are able to overhear Interest and Data packets related to other communications.

R-LF operates according to two phases: (i) a reinforcement learning that consists in maintaining a cost value for each available content prefix, (ii) adjustment of the waiting delay based on the neighborhood activity.

The first learning phase starts after a source node receives a randomly flooded Interest, and acts as follows: (i) the source node responds with a Data packet carrying the initial cost (e.g. 0), (ii) the first forwarder on the source-consumer path computes its cost with a reinforcement technique, replaces the C-field with the value obtained, and retransmits back the Data packet, (iii) each node on the path follows the same procedure until the Data packet reaches the consumer, (iv) in the vicinity of the path, the nodes that overhear the Data packet can perform a passive cost update to learn their eligibility relative to the data source.

The random flooding step is then over, and the first learning phase is set up. Each node maintains the cost related to the corresponding content prefix after retrieving (or overhearing) a Data packet with a smaller cost. Let us refer to this phase as the *reinforcement* phase.

To describe the forwarding process, we define the delay to wait before retransmission as $\Phi(a)$. The formal definition of $\Phi(a)$ will be detailed later. The forwarding decision in a relay node consists in finding the appropriate value of a that gives a correct delay to wait. Since a is calculated in two steps, let $a = \Delta + \theta$, with Δ and θ as explained in the following.

Let $C_x(p)$ and $C_y(p)$ be the current cost for prefix p at nodes x and y respectively. Whenever node x receives an Interest issued (or forwarded) by node y, it computes the value $\Delta = C_y(p) - C_x(p)$. Here, Δ quantifies the global eligibility of node x to forward the Interest. If $\Delta \geq 0$ then node x can potentially forward the Interest.

The value of θ is locally computed by the forwarder based on its neighborhood activity to refine Δ before calculating the delay time. Let us refer to this phase as the *Delta adjustment* phase. With the delta adjustment, the random-delayed forwarding is used only when the prefix is unknown by the forwarder (x) and is reset by the sender (y). Thus, even after an Interest timeout in x and y, the value of θ can be used in most cases to distinguish nodes eligibility.

After computing a, the Interest retransmission is delayed for $\Phi(a)$ units of time. During the delay-listening time, if node x detects that a forwarder z is transmitting a packet with the same name, it deduces that z is more eligible to handle the Interest and cancels its retransmission.

The next subsection provides the mathematical details.

5.3. Details and mathematical formalism

Reinforcement phase: The cost value at node x is updated according to the following reinforcement equation 1:

$$C_x(p) = (1 - \alpha)C_x(p) + \alpha\,(r + \min C_y(p)) \tag{1}$$

In this equation used in Q-learning and Q-routing [16], α is the learning rate, r is the reward, $C_x(p)$ is the cost at node x for the prefix p, and $\min C_y(p)$ is the smallest cost heard by node x from node y.

Assuming the hop-count as a metric, the reward is always equal to *1*, and the cost of each node increases as the distance to the content source increases. The cost at the content source is *0*.

The cost values reflect the distance to the content source, and are used to decide on a forwarder's eligibility. Moreover, the approximate nature of the update formula produces a large number of possible cost values over the network, which helps to avoid obtaining the same waiting delay for different nodes (Section 6.2).

Since the nodes remember only the smallest heard cost, a node may have an obsolete estimation of its cost value. To avoid that, after an Interest timeout, a node resets its cost value (i.e. $C_x(p)$) to *0* and the smallest heard cost (i.e. $\min C_y(p)$) to $\hat{\Delta}$, in order to accept cost updates. Note that in Interest packets, a cost value of *0* indicates that the sender has reset its cost or it has no information about the content prefix. Thus, it does not interfere with the *0* cost value of a Data packet that actually means that the packet has been sent by the producer. The estimation of $\hat{\Delta}$ is presented further.

Delta adjustment: The adjustment serves two purposes: it refines Δ to deal with local uncertainty in real time, and allows each node to handle multiple content prefixes simultaneously as it is shown in the next Section. In fact, using

only Δ to compute delays, even if it is accurate, does not allow different content names cohabitation to be supported.

To compute θ, let N_a be the neighborhood activity rate for all data names. From the perspective of a node, N_a can be computed by $N_a = D_u/I_d$, where D_u is the number of unsolicited received Data and I_d is the number of non-forwarded Interests (dropped Interests).

Then, θ may be simply defined as

$$\theta = Th - N_a \tag{2}$$

where $Th \leq 1$ is the activity threshold above which the waiting time should be increased. For simplicity, but without losing accuracy, N_a is kept between *0* and *1*. Thus, if Th is lower than *1* (e.g. *75%*), θ can be negative. In this case, the value of Δ is reduced, which will increase the waiting time. When no statistic is available, $N_a = Th$.

Delay function: After defining the appropriate value of a, the delay time is computed with a function that is inversely proportional to the value of a. Such a function can be intuitively defined by:

$$\Phi(a) = \frac{M}{e^{a/2}} + m \tag{3}$$

This function ensures that when two nodes can both forward an Interest, the node with the highest value of a will delay its transmission for a shorter time than the node with the lowest value, as depicted in Figure 3. Here, m forces the forwarder to wait for a minimum time to let the transmission of the corresponding Data packet if any.

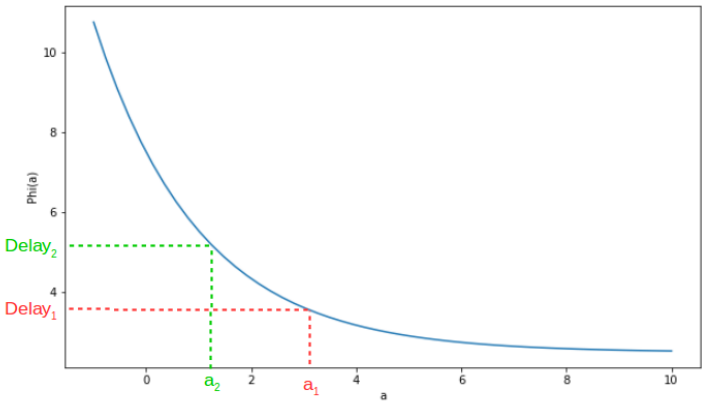


Figure 3: Delay function example

The importance of Φ is capital and a parameter calibration is needed to have an efficient distribution of waiting times. We can observe that $\lim_{a\to\infty} \Phi(a) =$

m. Therefore, we need to ensure that $\Phi(\hat{a}) > m$, where $\hat{a}$ is an estimation of the highest value of a.

According to Eq.2, θ is $\leq Th$. Given that the lowest cost value that can be received by a node is θ, we can deduce the highest gap between two cost values by estimating the maximum value of Δ. For that, we use the following property of the Q-learning formula:

$$C_x(p) \leq r + C_y(p) \Rightarrow C'_x(p) \leq r + C_y(p) \tag{4}$$

where $C'_x(p)$ is the updated value of $C_x(p)$. This property can be easily verified using the Q-learning update formula and initial conditions.

Assuming that in a network of n nodes, the maximum distance is $\sqrt{n}$ hops, we use Eq.4 to recursively estimate the maximum expected value of Δ as $\hat{\Delta} = (\sqrt{n} + 1)$. Then, we deduce an estimation of $\hat{a}$ to set the parameters of $\Phi(a)$.

The forwarding decision process for a node x with a cost of $C_x(p)$, receiving an Interest from node y with a cost of $C_y(p)$ under a prefix p is summarized in Algorithm 1.

Function ProcessInterest
Data: Interest packet
begin
  if *p is unknown* **then**
    if $C_y(p) == 0$ **then**
      Broadcast with random delay
    else
      Drop Interest (node not eligible)
    end
  else
    if $C_y(p) == 0$ **then**
      $\Delta = \hat{\Delta} - C_x$
    else
      $\Delta = C_y - C_x$
    end
    if $\Delta \geq 0$ **then**
      $\theta = Th - N_a$
      $a = \Delta + \theta$
      Broadcast with $\Phi(a)$ delay
    else
      Drop Interest (node not eligible)
    end
  end
end

Algorithm 1: Interest forwarding process

6. Evaluation

For evaluation purpose, we implemented the NDN module in the OMNeT++ simulator [17] including R-LF, BF and RONR forwarding strategies[2]. We first study the impact of the learning rate (α) on R-LF performance, and we fix the values for parameters M, N and Th. After that, we evaluate R-LF in comparison with RONR, a completely unicast approach, and BF which is a broadcast only mechanism that uses delayed retransmissions (Section 3).

The evaluation comparison studies 3 configurations: (i) a variable number of simultaneous data flows, (ii) one data flow with multiple consumers and cache enabled in relay-nodes, (iii) multiple data flows with a variable producer mobility speed. Finally, some experimental measures are reported on the R-LF implementation on the Arduino platform over IEEE802.15.4.

6.1. Simulation design

We consider a scenario that reflects a small-scale IoT monitoring application. The network topology is inspired from mobile WSNs: It consists of *16* fixed routers organized in a grid of *200m* x *200m*, through which producer and consumer nodes move following the Random Way Point mobility model. The MAC layer configuration reflects the IEEE802.15.4 properties (i.e. bit-rate, communication range and CSMA). Figure 4 gives an example of the simulated topology, and Table 2 reports the relevant simulation parameters.

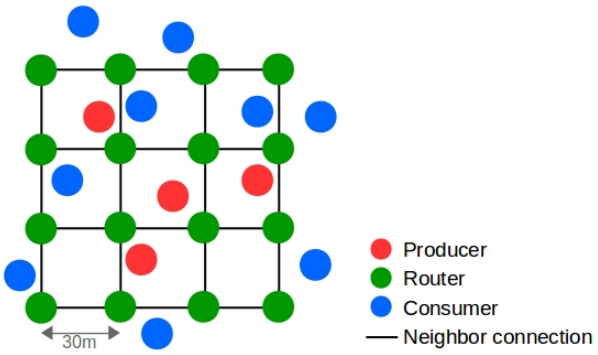


Figure 4: Topology example

[2]Source code available on `https://github.com/amar-ox/NDNOMNeT/tree/dev/main`

Table 2: Simulation parameters

Parameter	**Value**
Data packet size	*100 B*
Interest packet size	*50 B*
Interest send interval	*1 s*
Interest lifetime	*2 s*
Max Interest retransmissions	*1*
Cache size	*20* packets
Cache replacement	FIFO
Data freshness	*10 s*
Wireless bit-rate	*250 Kbps*
Wireless MAC potocol	*802.15.4 CSMA*
Communication range	*35 m*

In all the simulations, the reported results correspond to the average values obtained with 10 random executions. The following metrics are measured:

- The total number of transmitted packets, to study the accuracy of the forwarding decisions and the generated overhead.
- The mean round-trip time (RTT) for an Interest-Data exchange.
- The Interest satisfaction rate to show the global efficiency of the approach.
- The mean hop count of received Data, to study the ability of the strategy to find the nearest source (producer or cache) for the requested content.

6.2. Impact of the learning rate

R-LF involves a set of parameters that need to be calibrated to provide the best performance. Empirical simulation observations give the parameter values as follows: *M=5*, *N=2.5* and *Th=0.75*. However, the most important parameter to calibrate is the learning rate (α). To study the impact of α on R-LF performance, we simulate a scenario in which two consumers request the same content served by one producer, with *20* packets caching enabled in relays nodes. The results are presented in Figure 5.

Small values of α (i.e. ≤ 0.6) globally give poor performance. This can be explained by the fact that small values of α prevent nodes from learning fast by slowing down the update process. Likewise, when $0.9 < \alpha \leq 1.0$, the nodes overwrite previously learned values, which leads to more similar values and thus to more frequent collisions.

We find that $0.8 \leq \alpha \leq 0.9$ generally gives the best performance results. Moreover, these values provide stable results through the different executions. For this reason, even if $\alpha = 0.9$ and $\alpha = 1.0$ give good performance, $\alpha = 0.8$ is more likely to keep the same performance through the different executions. We conclude that *0.85* is a good value to obtain a satisfactory performance for the studied metrics. The value of α is fixed to *0.85* for the rest of the simulations.

Figure 6 shows the distribution of the values of a over the time for the *16* relay nodes. Globally, only a few similarities are observed in the values and the range of the possible values is completely exploited by the nodes. This is

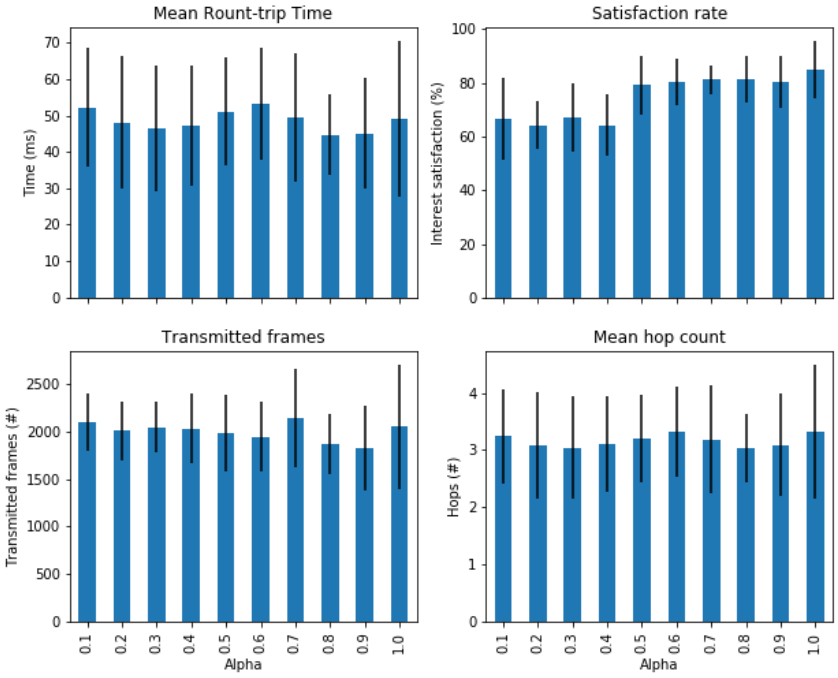


Figure 5: Impact of the learning rate

due to the reinforcement formula and the delta adjustment that helps to reduce similarities in the computed values. More similarities in a-values are observed between *-0.5* and *1.5*. However, according to the Φ formula (Figure 3), the delays corresponding to this range are high enough to let a node with a higher value of a retransmit first.

6.3. Multiple data-flows

The forwarding strategy has to support multiple namespace cohabitations. To create multiple data flows, each pair of consumer-producer exchanges packets under one name prefix. For example, consumer *C1* requests content created by producer *P1* under the prefix */ndn/home/humidity/**, and consumer *C2* requests content created by producer *P2* under the prefix */ndn/home/temperature/**. Practically, this case may occur when data is collected according to its type, or when multiple applications run in the same wireless network. Since only one consumer is requesting contents from one producer, caching is disabled in all nodes for this simulation. Figure 7 shows the metric measured according to 2, 4 and 8 simultaneous dataflows, which we believe is reasonable in a local monitoring application.

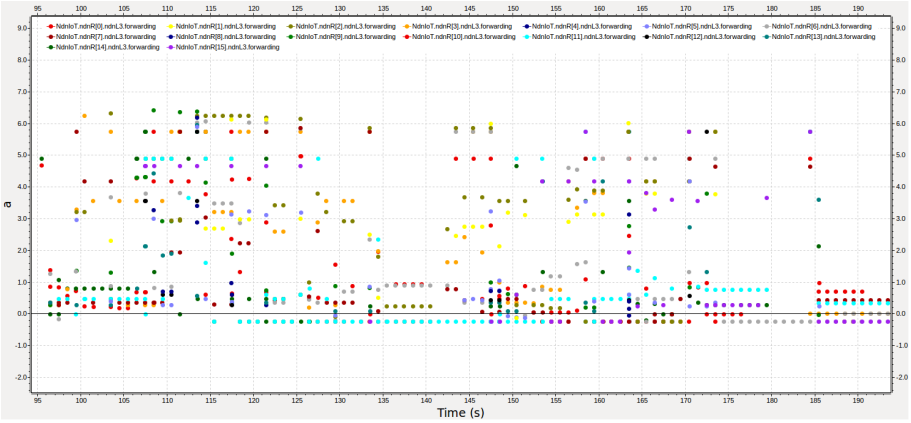


Figure 6: a-values example

The total number of frames transmitted by R-LF is very low, given that it is a broadcast-only technique. We observe that the number of frames transmitted with R-LF is much more close to RONR (unicast) than BF (broadcast). The low number of frames is the result of the Interest forwarding controlled by the reinforcement learning, followed by the delta adjustment step which can increase the waiting delay if the neighborhood activity is high. Moreover, the same difference between the three approaches is observed for 2, 4 and 8 data flows.

The satisfaction rate is roughly the same for the three approaches. However, the broadcast strategies (i.e. R-LF and BF) seem to be more adapted to support more simultaneous dataflows.

On average, R-LF and BF retrieve contents with almost one hop lower than RONR. Indeed, the unicast forwarding approach needs the expiration of the FIB entry to update the next hop address, while the broadcast forwarding can retrieve contents from any potential source in the vicinity of the forwarder. However, we can observe that this performance is achieved by BF at the cost of a large overhead, while R-LF gives the same efficiency with an overhead close to RONR.

R-LF provides an RTT lower than BF as it uses proportional timers instead of random ones. Moreover, R-LF scales better with more nodes/dataflows and becomes closer to RONR in terms of RTT. Obviously, RONR achieves the lowest RTT as forwarders do not use any delays before retransmitting. Having a reduced RTT is the main advantage of using a unicast strategy in this configuration.

6.4. Multiple consumers

In this case, only one content prefix (e.g. */ndn/home/**) is involved in the network. Multiple consumers simultaneously request contents provided by one

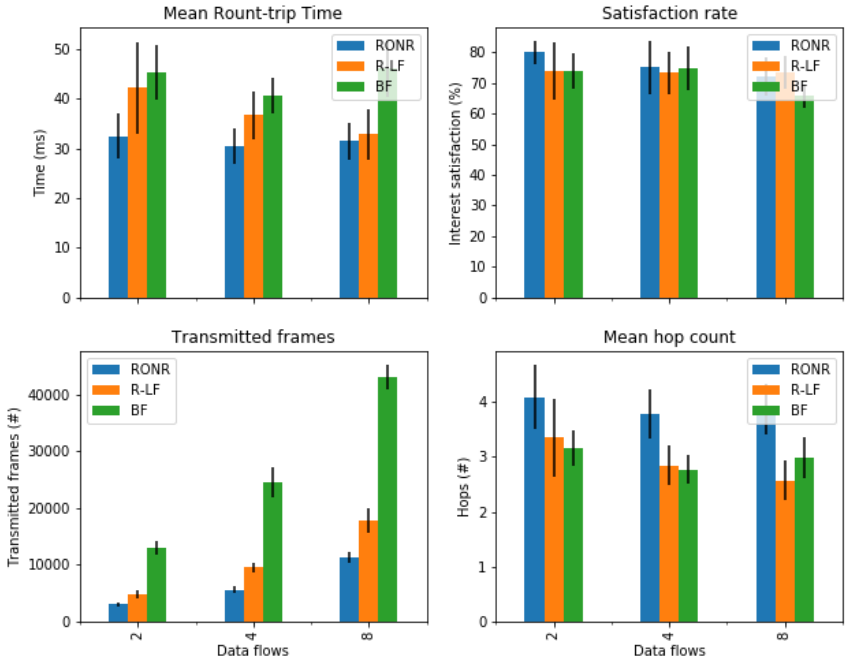


Figure 7: Multiple dataflows scenario results

producer, and/or by relay-node caches. This case is much more close to a many-to-one communication, such as a gateway-devices communication. However, it can also be considered as many-to-many due to caching that allows any relay-node to be a partial source of a content.

Figure 8 shows the results obtained for 2, 4 and 8 consumers.

Broadcast approaches (i.e. BF and R-LF) make better use of caching and achieve the highest satisfaction rate. The difference in the Data hop count is still in advantage of R-LF and BF. However, R-LF keeps its reduced overhead in comparison to BF.

When enabling caches, BF and R-LF are clearly penalized by the RTT in comparison to RONR. This is certainly due to the high competition access to the medium as the number of content sources increases with caching. Here, accurate Interest broadcast and delayed retransmissions of R-LF are not sufficient to provide a low RTT.

6.5. Producer speed

This case is the same as the first simulation scenario, with *4* dataflows. To study the forwarding strategies support for higher mobility, we increase the

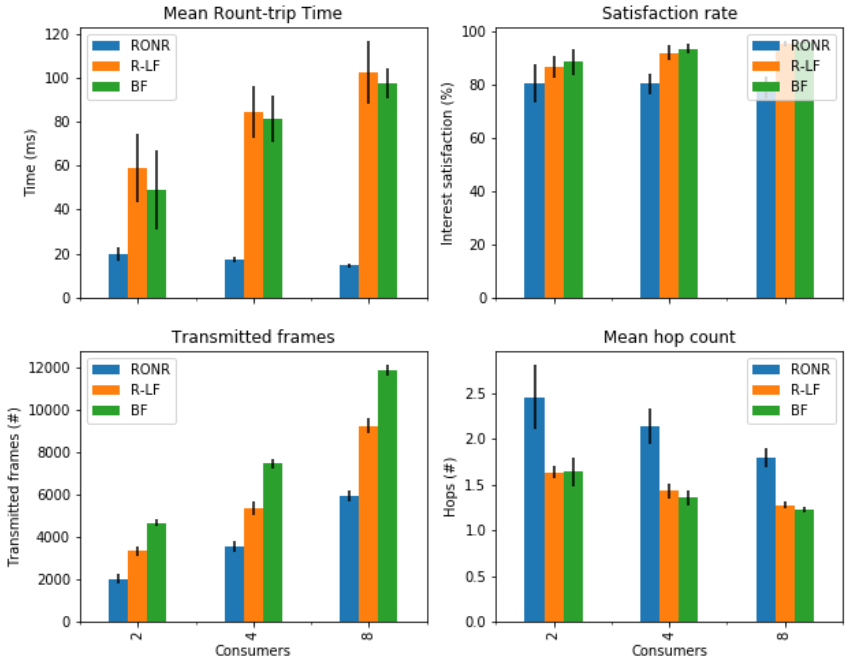


Figure 8: Multiple consumers and caching scenario results

producer speed from *1 mps* to *7 mps*.

Figure 9 shows the results obtained. As for the first scenario, the R-LF performance is close to the best among RONR and BF for each metric. We observe that when producers move faster, the RTT achieved by R-LF becomes closer to RONR, while it increases in the BF strategy. However, the satisfaction rate with RONR significantly decreases when the speed increases, while it decreases reasonably with R-LF.

Even under producer mobility, BF achieves a good satisfaction rate as broadcast is not much affected by the mobility. However, the high satisfaction rate achieved by BF comes along with a very large overhead, as shown by the total number of transmitted frames. Moreover, we observe that the number of frames transmitted in R-LF is slightly higher than RONR, but very low in comparison to BF. Finally, the shortest path is still ensured by R-LF even with a higher speed.

The unicast mapping shows its limitations to keep a good performance in presence of mobile nodes, while R-LF is able to provide a satisfactory performance.

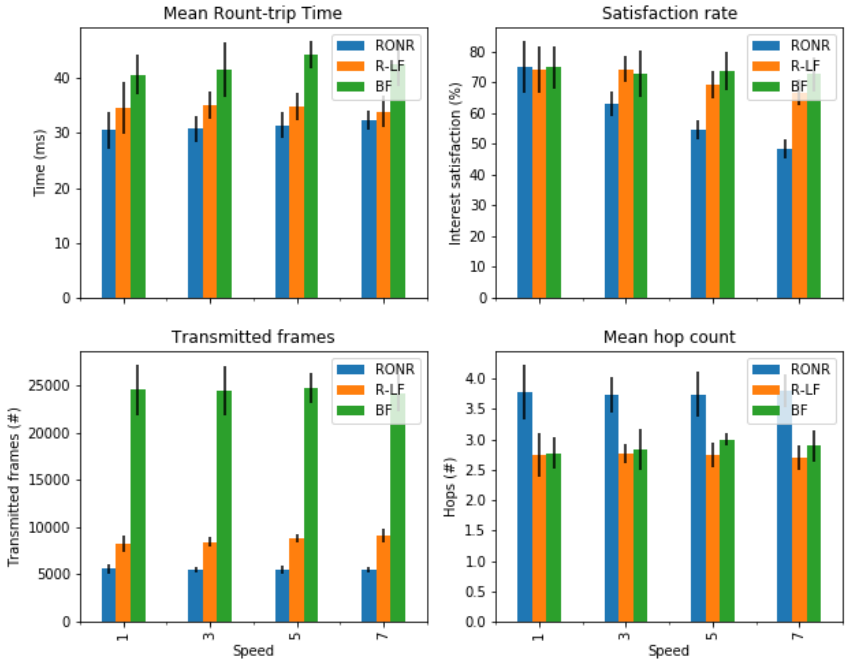


Figure 9: Producers speed scenario results

6.6. Implementation on IoT devices

We ported a simple version of the NDN module with R-LF on the Arduino platform[3]. We measured the time required to forward Interest and Data for both Arduino UNO (*16 Mhz*) and DUE (*84 Mhz*). The results are reported in Table 3. The Interest forwarding process for an unknown content name and an empty FIB (i.e. the first Interest) takes approximatively $145\mu s$. The Interest forwarding process for a known content name, with *5* FIB entries, is about $50\mu s$. This difference is mainly due to the random number generation used for the first Interest flooding. A FIB entry update after receiving a Data packet takes $70\mu s$.

On the Arduino DUE, the time required for the first Interest forwarding is $55\mu s$, and $10\mu s$ when the content name is known, considering 10 FIB entries. The FIB entry update time is about $18\mu s$.

Concerning the memory space required by the implementation, only 28% of storage is needed for the UNO and 50% of RAM. The implementation on the Arduino DUE occupies 6% of the total memory. The implementation includes

[3]Source code available on `https://github.com/amar-ox/NDN802.15.4-arduino`

a simple NDN packet definition, the three NDN data structures and the R-LF process over IEEE802.15.4.

Obviously, these values will increase when security algorithms will be added. However, the leeway is still important for security and reliability improvements. As an empirical comparison, some open source implementations of the IPv6 stack over IEEE802.15.4 take about 12% storage and 45% RAM in the Arduino Mega[4], while R-LF takes 3% storage and 12% RAM.

Table 3: R-LF measures on Arduino

Operation	**Time (UNO)**	**Time (DUE)**
First Interest forwarding	$145\mu s$	$55\mu s$
Subsequent Interest forwarding	$50\mu s$	$10\mu s$
FIB entry update	$70\mu s$	$18\mu s$

7. Discussion and conclusion

In the three simulated scenarios, the R-LF performance is always close the best one among RONR and BF for all the metrics. This shows the adaptability of R-LF to different communication scenarios. The only exception to this observation is the RTT in the multi-consumer scenario, which certainly requires more sophisticated techniques for retransmitting cached Data packets.

In our strategy, the random Interest flooding occurs rarely, as the delta adjustment step can be used to compute a waiting delay even if the cost value is reset. This contributes to significantly reduce the network overhead and represents an important improvement regarding the basic *flood-and-learn* schema.

The reinforcement learning provides a satisfactory performance without requiring an explicit exploration phase. Hence, to forward an Interest, a node needs only a simple FIB look-up as expected in NDN.

The R-LF strategy supports complex communication patterns. The results obtained through different scenarios show that the accuracy of the unicast mapping can be approached by a broadcast-based strategy. Moreover, they show the adaptability of the R-LF approach to support multiple dataflows, multiple consumers and nodes mobility.

As a future work, we aim to improve the overhearing efficiency and media access for cached Data retransmission, especially in multi-consumer cases.

Acknowledgments

This work is supported by the 17MDU988 project which is a PHC (Partenariat Hubert Curien) Tassili, a French-Algerian cooperation program. This Tassili project is funded by the French ministries (MEAE and MERI) and the Algerian ministry MESRS.

[4] `https://github.com/telecombretagne/Arduino-IPv6Stack`

References

[1] I. Yaqoob, E. Ahmed, I. A. T. Hashem, A. I. A. Ahmed, A. Gani, M. Imran, M. Guizani, Internet of things architecture: Recent advances, taxonomy, requirements, and open challenges, IEEE Wireless Communications 24 (3) (2017) 10–16. doi:10.1109/MWC.2017.1600421.

[2] W. Shang, Y. Yu, R. Droms, L. Zhang, Challenges in IoT networking via TCP/IP architecture, Tech. Rep. NDN-0038, NDN (February 2016).

[3] B. Ahlgren, C. Dannewitz, C. Imbrenda, D. Kutscher, B. Ohlman, A Survey of Information-Centric Networking, IEEE Communications Magazine 50 (7) (2012) 26–36.

[4] L. Zhang, A. Afanasyev, J. Burke, V. Jacobson, kc claffy, P. Crowley, C. Papadopoulos, L. Wang, B. Zhang, Named Data Networking, ACM SIG-COMM Computer Communication Review 44 (3) (2014) 66–77.

[5] W. Shang, A. Bannisy, T. Liangz, Z. Wangx, Y. Yu, A. Afanasyev, J. Thompsonx, J. Burkex, B. Zhangz, L. Zhang, Named Data Networking of Things (Invited paper), in: The 1st IEEE Intl. Conf. on Internet-of-Things Design and Implementation, Berlin, Germany, 2016.
URL https://named-data.net/publications/ndn-iotdi-2016/

[6] E. Baccelli, C. Mehlis, O. Hahm, T. C. Schmidt, M. Wählisch, Information centric networking in the iot: Experiments with ndn in the wild, in: Proceedings of the 1st ACM Conference on Information-Centric Networking, ACM-ICN '14, ACM, New York, NY, USA, 2014, pp. 77–86. doi:10.1145/2660129.2660144.
URL http://doi.acm.org/10.1145/2660129.2660144

[7] W. Shang, Y. Yu, R. Droms, L. Zhang, Challenges in IoT networking via TCP/IP architecture, Tech. Rep. NDN-0038, NDN (February 2016).

[8] M. Amadeo, C. Campolo, A. Molinaro, Forwarding strategies in named data wireless ad hoc networks: Design and evaluation, Journal of Network and Computer Applications 50 (Supplement C) (2015) 148 – 158. doi:https://doi.org/10.1016/j.jnca.2014.06.007.
URL http://www.sciencedirect.com/science/article/pii/S1084804514001404

[9] D. Saxena, V. Raychoudhury, N. Suri, C. Becker, J. Cao, Named data networking: A survey, Computer Science Review 19 (2016) 15 – 55. doi:https://doi.org/10.1016/j.cosrev.2016.01.001.
URL http://www.sciencedirect.com/science/article/pii/S1574013715300599

[10] J. Shi, E. Newberry, B. Zhang, On broadcast-based self-learning in named data networking, in: 2017 IFIP Networking Conference (IFIP Networking) and Workshops, 2017, pp. 1–9.

[11] L. Wang, A. Afanasyev, R. Kuntz, R. Vuyyuru, R. Wakikawa, L. Zhang, Rapid traffic information dissemination using named data, in: Proceedings of the 1st ACM Workshop on Emerging Name-Oriented Mobile Networking Design - Architecture, Algorithms, and Applications, NoM '12, ACM, New York, NY, USA, 2012, pp. 7–12. doi:10.1145/2248361.2248365.
URL http://doi.acm.org/10.1145/2248361.2248365

[12] M. Michael, P. Vasileios, Z. Lixia, Listen first, broadcast later: Topology-agnostic forwarding under high dynamics, in: Annual conference of international technology alliance in network and information science, 2010.

[13] Y. T. Yu, R. B. Dilmaghani, S. Calo, M. Y. Sanadidi, M. Gerla, Interest propagation in named data manets, in: 2013 International Conference on Computing, Networking and Communications (ICNC), 2013, pp. 1118–1122. doi:10.1109/ICCNC.2013.6504249.

[14] P. Kietzmann, C. Gündogan, T. C. Schmidt, O. Hahm, M. Wählisch, The Need for a Name to MAC Address Mapping in NDN: Towards Quantifying the Resource Gain, in: ACM ICN 2017 - 4th ACM Conference on Information-Centric Networking , Berlin, Germany, 2017.
URL https://hal.inria.fr/hal-01666601

[15] Y. Zhang, M. Fromherz, Constrained flooding: a robust and efficient routing framework for wireless sensor networks, in: 20th International Conference on Advanced Information Networking and Applications - Volume 1 (AINA'06), Vol. 1, 2006, pp. 6 pp.–. doi:10.1109/AINA.2006.132.

[16] J. A. Boyan, M. L. Littman, Packet routing in dynamically changing networks: A reinforcement learning approach, in: Proceedings of the 6th International Conference on Neural Information Processing Systems, NIPS'93, Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1993, pp. 671–678.
URL http://dl.acm.org/citation.cfm?id=2987189.2987274

[17] OMNet++, OMNeT++ discrete event simulator, [Online]; https://www.omnetpp.org/.