A collaborative caching strategy for content-centric enabled wireless sensor networks

Ghada Jaber, Rahim Kacimi

A Collaborative Caching Strategy for Content-Centric enabled Wireless Sensor Networks

Ghada Jaber[a,*], Rahim Kacimi[b]

[a] *University of Technology of Compiegne, Heudiasyc, Compiègne, France.*
[b] *University of Toulouse, IRIT-UPS, 118 route de Narbonne, 31062 Toulouse, France.*

Abstract

The Content-Centric Networking (CCN) is an efficient traffic handling technology by accessing content with its name instead of its physical location and achieving in-network caching. Indeed, CCN caching ensures high content availability, network traffic reduction, and low retrieval latency which reduces congestion and improves end-to-end delay. Moreover, it could greatly improve the efficiency of content delivery in Wireless Sensor Networks (WSNs). In this article, we propose to exploit this feature to enable CCN in WSN environments. The CCN architecture enables the content caching on each sensor-node in WSN and several research studies have been devoted to the caching management issue in such a context. However, caching the content on all the nodes is not a good strategy in terms of resource utilization. It is, therefore, necessary to determine where to cache and how to handle it in order to optimize the resources while realizing a high-interest satisfaction rate. Thus, our objective is to study existing caching strategies and propose a novel one that takes into account the node centrality and its distance from the source of the content. Through extensive simulations, we examine the performance of our scheme under different configurations and demonstrate how it outperforms the traditional caching strategies such as LCE (Leave Copy Everywhere) and LCD (Leave Copy Down).

Keywords: Wireless sensor networks, information centric networking, content-centric networking, caching, energy efficiency, content popularity.

1. Introduction

Wireless sensor networks are an essential part of the "perception" layer of the Internet of Things (IoT). They connect the digital world created by conventional computer networks to the physical world. Moreover, they continually bring new applications to out lives through a large number of nodes that collect, process and disseminate environmental data. Therefore, today circulates on this layer a large and varied volume of data generated in a continuous way with a greater emphasis on the information and not on its source location [1].

Besides naive data dissemination approaches such as flooding and gossiping, many communication patterns are being studied in wireless sensor networks and they do not often follow the sender/receiver one of the Internet or the many-to-one pattern of telemetry and monitoring applications. Indeed, there are several applications for which traditional schemes does not fit. For instance, data-centric paradigms wherein a sensor node requests information where an interesting event is hapenning (e.g., sensor reading has exceeded a threshold). Thus, the requester does not necessarily wish to request information from a particular node but rather from any node that can provide the data. Similarly, it is often the case in Publish-subscribe approach which allows a sensor to publish the data readings it produces. Then the other nodes may subscribe to the data by registering an interest.

Furthermore we can find CCN which offers in-network caching that contribute in alleviating the pressure on the network bandwidth in WSNs while spreading content copies between the network nodes in a distributed and efficient manner [2, 3, 4]. Caching is the essential feature of content-centric networking which has been used for many enhancement like fault-tolerance, improving communication over WSNs, multicasting applications, and improving the network performance [5]. Thanks to this feature, users can recover faster the requested content from the intermediate nodes [6]. Then, the traffic load could significantly be reduced and the data availability could increase [7]. However, it is important to ensure that the adopted in-network caching has to be efficient and manage content distribution in an intelligent way. In-network caching in ICN depends essentially on the caching strategy which identifies the content placement and on the cache replacement policy that decides on which content to eject from the cache once this latter is full. Therefore, these two strategies have to be well investigated in order to fit the WSNs requirements.

Our contribution is related to the in network-caching. Our goal is to study the existing caching strategies (placement and replacement ones) and try to see the impact of certain parameters by proposing a caching strategy which takes into account the network constraints in order to optimize the network performance. Indeed, to ensure a certain level of diversity, cache hit, and energy efficiency, we propose an effective caching placement strategy that chooses the sensor-nodes on which to cache by taking into account their degree and their distance from the source of the content. We also proposed a scheme to avoid

*Corresponding author. Tel.: +33 561-557-471.
Email address: `ghada.jaber@utc.fr` (Ghada Jaber)

as possible the problem of interest loop in CCN-WSNs which causes energy wasting and latency increasing. We also compare the proposed strategy to the existing approaches to show how it overcome them on certain performance metrics.

The remainder of this article is organized as follows: in the next section, we survey the recent work on in-network caching and CCN. Section 3 presents the use case. The proposed strategy is described in section 4. Section 5 details the performance evaluation part. In section 6, we summarize and conclude this study.

2. Related work

In-network content storage has become an inherent capability of content-centric networking architecture which raises new challenges in the use and the provision of the caching placement. Therefore, a trade-off between the network performance and the provisioning cost has to be found [8]. As one of the most promising potential architecture for the future Internet, Content-Centric Networking has attracted a lot of attention. CCN integrates the content routing and caching process in the network layer, and can potentially achieve good performance in terms of resource utilization and content dissemination efficiency [9]. Hence, it offers transparent and ubiquitous in-network caches which are the fundamental building block that guarantees efficient content retrieval [10]. Besides, caching nodes in CCN are application-independent. By against, the traditional caching system is application dependent and caches use proprietary protocols [11]. Among several existing works in the literature, it was demonstrated in [12] that ICN caching mechanism including CCN outperforms traditional caching technologies in terms of network quality of service especially delivery latency, network resource consumption, and end-user experience. Nevertheless, in-network caching poses many decision challenges related to "content placement" (WHERE to cache the content), 'content replacement' (WHICH content is to evict from the cache), and 'request routing' (HOW to redirect interests to an optimal cache).

Caching data at locations that minimize the number of transmissions in the network reduces the power consumption in the network and hence extends its lifetime. However, excessive caching can lead to high costs (in terms of energy and delay) and performance degradation. Finding the best locations of the nodes for caching optimizes the communication [13]. Effectively, caching in CCN deals with several types of traffic and any node in the network can handle caching. However, old caching technologies are defined for specific traffic and usually located in a predetermined location such as the LCE strategy (Leave Copy Everywhere) and LCD strategy (Leave Copy down) [14]. Indeed, LCE is defined by its operation of caching data in every node crossed. Part of the practice of the ICN is the ability to make information readable and easily accessible as described in ICN initial proposal. When LCE is used, one the user sends an interest, the cache hit is high due to the success of finding content available in nodes.

LCE is generally a good choice in case of flash-crowd events or cases of highly skewed content popularity distributions and it does not require any coordination [15]. However, LCD is a cache management strategy that defines the form and manner of content caching on nodes. Its operation works similarly to the popular 'drop at the first neighbor' process. This technique requires minimal coordination among caching nodes as they can signal to other downstream nodes whether to cache the content or not.

Gayathri *et al.,* [16] proposed an information-centric scheme for wireless sensor networks using cognitive in-network devices. Then, the routing decisions became dynamic and based on specific Knowledge and Reasoning-observations in the WSNs. These techniques are used at the cognitive device and can provide reliability, better delay, and network throughput over the communication paths.

As discussed in [17], many researchers start to exploit information-centric networking in WSNs since they present the major technique in the sensing layer of the Internet of Things. The authors proposed a collaborative caching strategy for the information-centric wireless sensor network composed of three steps: the cache size adjustment based on the node betweeness, the cache decision which is based on the data replacement frequency and cache replacement policy based on content value. They argue that nodes can find a balance between caching performance and storage consumption.

Authors in [5] provided a model for the trade-off between multihop communication costs and the freshness of a transient data item. They showed that the model can successfully capture the effect of data transiency and realizes considerable savings in terms of reduction of network load, especially for highly requested data items.

In [13], the authors presented a cache placement scheme called NCP that takes into consideration IoT traffic. When selecting the optimal caching nodes, the proposed scheme minimizes the cost of moving the content from source nodes to intermediate nodes, the cost of content caching and the cost of content delivery to users. It shows through a multiobjective function that the cache utilization was enhanced with fast data retrievel, diversity and cache distribution.

The design of cache replacement algorithms is realized for content distribution purposes. When the network becomes stable and the node cache overflows, a replacement policy, such as Least Recently Used (LRU), Least Frequently Used (LFU) or First In First Out (FIFO) is used to evict one of the cached contents to make room for the newly arrived one. Indeed, LFU statically places in the cache the C most frequently requested data [15] and its implementation requires content popularity ranking to be known beforehand. While in LRU, the idea is to keep the data recently used and to replace it with the other data. LRU has two advantages that make it very popular, it is very responsive to non-stationary trends since its replacement decisions are exclusively based on recency and it cannot perform significantly worse than LFU because the ratio between the optimal cache hit ratio and LRU cache hit ratio is bounded [15].

We assume that the content placement and replacement play a significant role in the resulting in traffic and energy reduction. Furthermore, the selection of an appropriate node so that it could be able to serve future requests for a longer time is very important. Consequently, addressing the location problem of caches is an important part of the campaign for in-network caching in CCN. We remind that, in this article, we consider sensor environments. Hence, we propose a CCN-WSNs context-aware caching strategy which integrates new content placement and replacement policies.

To summarize, many works investigated content caching in CCN. Several works treated this in CCN-WSNs. However, to the best of our knowledge, no one proposed a caching strategy that combines at the same time the node degree and its distance from the source, and aims to reduce energy consumption, traveled path and increase cache diversity. Motivated by the aforementioned shortcomings, this article proposes a design of a caching strategy to decide where to place content copies depending on the node degree and its distance from the source.

3. Use Case

In this section, we present some essential use cases in neO-Campus grant [18]. These scenarios raise the need for implementing CCN on sensor nodes and proposing new caching strategies that handle content dissemination and use energy efficiently. In figure 1, we present the architecture of CCN in the WSNs of the campus. As we can see, CCN communication layer handles packet transmission and does not rely on other transport protocols to deliver messages. The communication model is built entirely and uniquely on named data [19]. When integrated in WSN, CCN binds a name to the data itself. A CCN-WSN is a lightweight alternative to implementing IP for sensor networks. Indeed, memory and communications constraints in WSNs are taken into consideration when meeting CCN concepts.

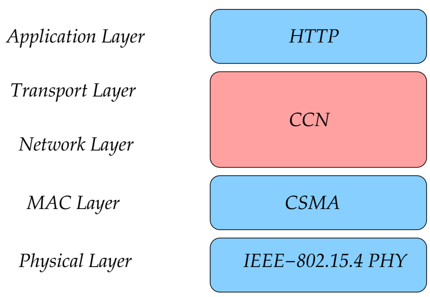


Figure 1: Architecture of CCN in WSNs.

In this paper, we present the example of Paul Sabatier university with many buildings equipped with sensor networks to build our use cases.

Every building, restaurant, parking and university library is equipped with sensors that measure different pieces of information. For example, sensors at classrooms measure the temperature, the humidity, luminosity, presence, position of professors, energy equipment consumption and contain information about the schedule. Students and staff use their devices to ask for contents and they are interested in getting the requested content without having an idea about its location.

As shown in fig. 2, several scenarios can be gathered in a smart campus. Let us consider a 1^{st} Wireless Sensor Network ($WSN1$) operating in the building U4 of the university, a 2^{nd} sensor network ($WSN2$) operating in the restaurant and 3^{rd} one ($WSN3$) operating in the university parking. Other sensor nodes are randomly widespread on the campus to ensure communication between different sensor networks.

Scenario 1: Scenario 1: A user (e.g., student) on the campus is interested in the temperature in the classroom 204 in building $U4$ operated by $WSN1$. He broadcasts an interest in the network and the node with the corresponding content replies with it. The sensors respond to the queries of each user and the content is cached on the nodes that are on the response path. Then, whenever and wherever, a student or a personal staff wants to know something about classrooms, he can send a query.

Scenario 2: The sensor network $WSN2$ in the restaurant affords information about the state of the restaurant. When students are interested in the state of the queue in the restaurant and the menu, in the same manner, they send interests requesting information and the sensors measuring this information reply.

Scenario 3: $WSN3$ gives information about the availability of places in the parking. The campus is big and users may have information about this wherever they are on the campus.

The traffic is not the same the whole day. In the rush hours, for example, around 11h30, students are interested in the state of the restaurant. Between 7.30 am and 8 am, students arrive at the campus and try to find an available place in the parking. Other users look for professors during the break.

Users who ask for this information later get the response faster since the information was already broadcasted on the network and is available in intermediate nodes.

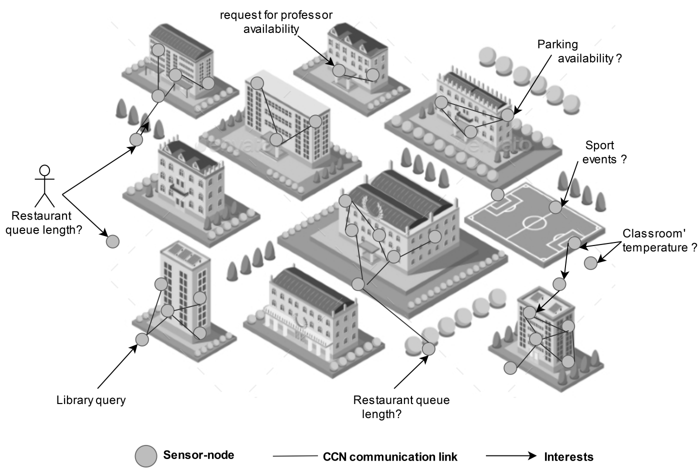


Figure 2: Example of a smart campus.

The future digital campus will be accessible, sustainable, and smart. It will be full of sensors, autonomous but capable of evolving. The challenge will be to intelligently manage their data. For these reasons, we thought about proposing a new caching strategy to manage efficiently this amount of network data on the campus.

4. Proposed strategy: A collaborative caching strategy for content-centric enabled wireless sensor networks

Before describing our design, we detail the interest distribution that we adopt in this article.

4.1. Content request process

We consider that interest packets follow the Zipf distribution which presents the frequencies of distribution of user interests in the network. This distribution assigns a rank for popular content. Popularity means that out of all available contents how many times a particular content is accessed. If the content is more popular then its rank is low and if the content is less popular then its rank will be high. Let $E = N_1, N_2, N_3, ..., N_{20}$ denote the content population in the system with a size of 20 contents. Since content popularity follows the Zipf distribution, the i^{th} most popular content is requested with a probability proportional to:

$$p_i = \frac{\beta}{i^{\alpha}} \tag{1}$$

where β is the normalized constant with $\beta = \frac{1}{\sum_{i=1}^{20} \frac{1}{\alpha}}$ and α is the Zipf exponent.

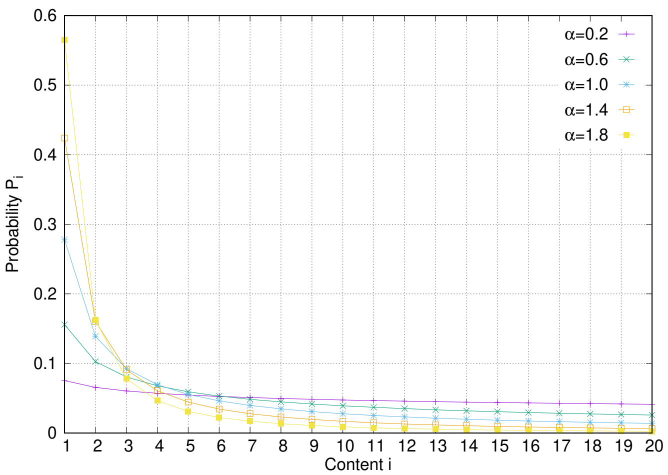


Figure 3: Zipf interest distribution depending on the value of α varied from 0.2 to 1.8 for 20 different contents.

Fig. 3 shows the probability distribution p_i for each content when α varies from 0.2 to 1.8. As shown, when the popularity is low ($\alpha = 0.2$), the probability is nearly the same for all the contents since they have almost the same popularity. Therefore, they behave like a uniform probability law where the probability to request content is similar. In addition to that, in many studies, researchers showed that $\alpha = 1$ refers to normal popularity where 90% interests request 60% of content. Once the α increases, the probability p_i increases for the most popular content. For instance, for $\alpha = 1.8$, the content $N1$ is requested with a probability equal to 0.58.

4.2. Cache admission control

In CCN, in the caching strategy area, as already stated, researchers basically try to develop algorithms that choose appropriate nodes to cache incoming content. CCN supports ubiquitous caching protocol where every node needs to cache incoming content. But sometimes ubiquitous caching is wastage of resource and it is not a smart strategy to cache the same content on each node. Hence, we need to find a caching strategy to appropriately select the node to cache newly arrived content.

When a sensor node receives the requested data or a data goes through it, a cache admission control is triggered to decide whether it should be stored into the cache of the node or not [8]. Inserting data into cache might not always be favorable because the incorrect decision can lower the probability of cache hit and also makes poor utilization of the limited storage. We firstly explored the Steiner Point [20] to find the best location to cache the content. For instance, in [20], they tried to minimize communication costs by finding the nodes of a weighted Minimum Steiner tree. In other words, they created a Steiner Data Caching Tree. They showed that the degree of the node where to cache is 3. Furthermore, they consider that the formation of Steiner data caching tree is done by considering the refresh rates in each edge of the tree. However, they addressed the scenario where multiple subscribers were receiving data from one source. This is not our case, since users may receive data from multiple nodes. Besides, in the considered scenario, the contents are pushed in the network once they are requested by users. Thus, we do not consider the concept of refresh rate (in this work, an on-demand scenario is considered).

As for [21], the authors contend that node degree is not an interesting insight to consider when caching because in a network of caches the consumer is interested in connecting to the content, not to a specific node. However, we argue that in a wireless network, a node with a high degree may guarantee content availability when it detains a high number of neighbors.

4.2.1. Proposed cache placement approach

For the reason cited above, we chose to stay more general and to consider the degree of the node. Hence, in the proposed strategy, the cache admission decision at a node is based on two conditions: (i) the percentage of the path from the source is it greater than Δ ? (ii) if yes, the node does it have a number of neighbors greater than x? The more the node has an important number of neighbors, the more nodes cache the content and ensure content availability in the network. Note that Δ is calculated as a function of the minimum number of hops and x as a function of the vicinity cardinal node.

Therefore our strategy is a '*collaborative Caching Strategy Distance and node Degree aware in content-centric enabled wireless sensor networks*' called CSDD.

A trade-off exists between query latency and content accessibility. With a small Δ, the number of copies for each content is high and access delay for this content object is low. On the other hand, with a larger Δ, each content has a small number of copies in intermediate nodes, and the access delay can be longer [22]. However, this depends on the position of the user.

Fig. 4 explains the first condition of the caching strategy and Δ variation; from how much Δ from the source, our strategy decides to cache? In the next section, we will investigate the impact of this parameter in the proposed caching strategy.

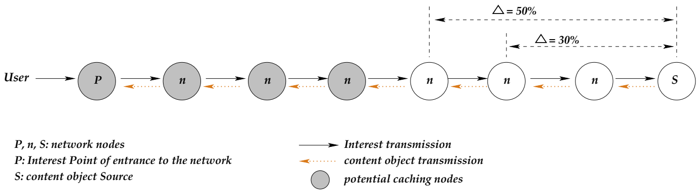


Figure 4: The first condition of our caching scheme: distance from the source node.

Once the first condition is fulfilled, our strategy checks the degree of the nodes on the path, if it is greater than x, the content is cached. Otherwise, the content is forwarded. If the second condition was not verified all the way, the content is cached in the edge node (near the user). For this, a small field is added to the content to check if it was cached on the path or not. When the content reaches the edge node, it checks if it was cached. If yes and if the edge node does not verify the second condition of the strategy, it is just forwarded. Differently, it is cached on the edge node without taking into consideration the verification of the second condition.

Loop avoidance

When implementing the caching strategy, we noticed that when replying by a content, since we are working on a wireless communication mode, some contents may be sent two times to a node. In a normal case, when a node receives a Content Object CO, if it does not have CO in its Content Store (CS), it caches CO, otherwise, it forwards CO. Since in our strategy, the content is not cached in all the nodes of the network. If a node does not cache the received content (because it does not verify the condition of the strategy), it just forwards it. The next nodes who receive the interest will also forward the interest (whether cached or not). Consequently, the nodes that already have received the content but did not cache it, when receiving interest for the same content, repeat the same process. Which causes a problem of loop hence a waste of energy consumption. To solve this problem, we proposed to give an ID to every generated interest and this to avoid treating the same interest for a defined time.

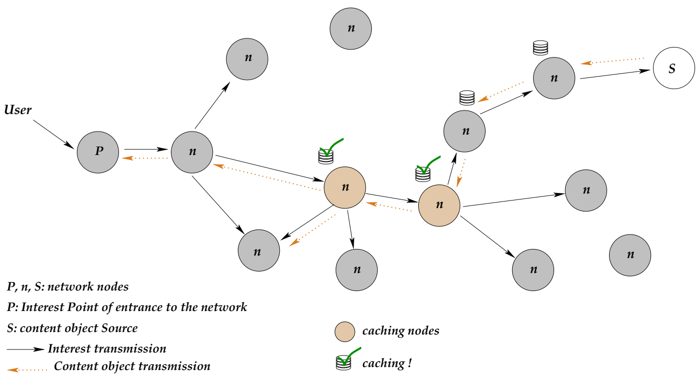


Figure 5: An example of function of our strategy when Δ is equal to 50% and $x > 3$.

Fig. 5 shows how the strategy works if a user sends an interest for a content and if Δ is equal to 50% and $x >= 3$. When the requested content object is found, it is sent back to the user. Meanwhile, it is cached in nodes that fulfill the two conditions related to the degree x and position from the source Δ. As already explained, if the second condition was not verified, the content would be cached in the first node near the user.

Algorithm 1: Proposed caching strategy CSDD

```
input : Δ: Distance from the node having the content
        n: Number of neighbors
        x: Threshold set for the number of neighbors
//A user send an interest I for a content co_i. Once
  the interest is received on a node, a counter is
  set. When the interest is forwarded from a node to
  another the counter is increased by 1.
Receive an Interest I;
Set counter t = 0;
while I is forwarded from a node to another do
    //t will help in calculating the distance crossed
      by the interest to attend the node having the
      corresponding content.
    t ++;
    if the content is found in the CS then
        //t corresponds here to 100% of the path.
        Assign t to the content;
        //d is the required distance (from the node
          having the content) from where our strategy
          decides to start caching.
        Calculate d = t*Δ/100;
        while the content is forwarded from a node to another do
            t - -;
            //Check the first condition of our scheme
              concerning the distance from the node
              having the content.
            if (t ≤ d) then
                //Check the second condition of our
                  scheme concerning the degree of the
                  node.
                if (n ≥ x) then
                    Cache the content;
                    Mark the content as already cached;
                    f = TRUE;
            if (t = 1) then
                if (f = FALSE) then
                    Cache the content on edge node;
```

4.3. Cache replacement policy

Our mechanism uses a weak consistency model ages travel by multiple hops, it is important to ensure high reliability. Once the CS is full, our replacement policy relies on replacing the less popular content in the node content store with the new content. It aims to keep popular contents in the CS. Then our replacement policy is Popularity-based. Moreover, the interests follow the Zipf distribution. It is also worth noting that the popularity is added to the content object and once this content is cached in the CS it is cached with its popularity.

In our replacement policy, we suppose that popularity denoted by P of content $N1$ is bigger than the popularity of $N2$. Then, $P(N1) > P(N2) > P(N3) > ... > P(Nn)$. Since the cache size is limited, the node only caches the most popular content

and evicts the less popular. If two contents of the same popularity exist when applying replacement, our strategy evicts the content having the smallest index for example if $N8$ and $N7$ have the same popularity $N7$ is evicted.
The proposed strategy including its cache placement and replacement mechanisms are detailed in the Algorithm 1.

5. Performance evaluation

Here, we examine the performance of CSDD under different degree values x and distance Δ. We then compare it to LCE and LCD, under two replacement policies FIFO and Popularity-based. Moreover, we varied the Zipf exponent α.

5.1. Simulation set-up

For the implementation of the existing strategies and the proposed scheme, we used the CCNx_Contiki framework [23] and we modify it to follow the requirements of all the strategies. CCNx enables the exportation of the code on real platforms. In the simulation, many users poll their interest through 4 entrance points in the network. Simulation parameters are detailed in 1
In this paper, we do not treat the mobility [24] of nodes so we consider that the network nodes are static. Besides, based on the use case, the nodes are randomly deployed. For the sake of simplification, we suppose that the contents have the same size. As already mentioned, the packets follow the Zipf distribution [25].
The simulations run 10 times using different random seeds and present the mean values as results. 95% confidence intervals (CIs) are calculated and the values are below 8.5×10^{-4}. Therefore, confidence intervals are not drawn on the figures since they are very low.

Table 1: table

Simulation Parameters

Simulation Parameters	**Value**
Area	$500 \times 500\ m^2$
Simulation duration	3600 s
Radio coverage range	100 m
Initial energy	2 J
Cache size	6 (30%)
Number of nodes	85
Number of generated types of content	20
Number of entrances on the network	4
Values of α	0.2, 0.6, 1, 1.4, 1.8

Since our strategy is based on the node degree, we study the percentage of nodes that have a certain number of neighbors and we plot the results in Fig. 6. Then, fig. 6 represents the percentage of nodes detaining different degrees(cardinal of its vicinity). As we may notice, more than 57% of the node network presents a degree higher than 2 and 16% of nodes detains a degree higher than 3.

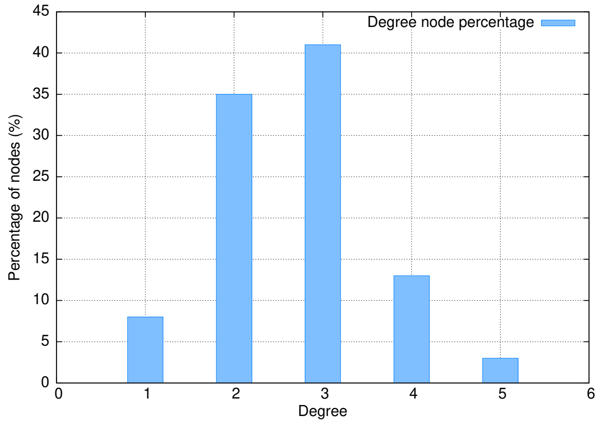


Figure 6: Percentage of nodes detaining a certain node degree.

5.2. Evaluation metrics

In this article, evaluation metrics are energy consumption, network lifetime, stretch, cache diversity and cache replacement rate.

In order to evaluate the proposed strategy, we continue to evaluate the energy consumption under the model proposed in [26]. The notion of network lifetime chose in this study presents the duration until the first node exhausts all its energy [12].

As already mentionned, we also present results for the Stretch which defines the percentage of the path that has been crossed to retrieve the content [27].

$$Stretch = \frac{\sum_{i=1}^{I} hops\_crossed_i}{\sum_{i=1}^{I} total\_hops_i} \tag{2}$$

Where I defines the number of total generated contents ie. all measured contents by sensors

The next metric is the cache Diversity which measures the number of distinct elements stored in the caches. It expresses the ratio between the cardinality of unique contents stored in all caches and the cardinality of total number of contents in the caches [27].

$$Diversity = \frac{Card \bigcup_{n=1}^{N} CO_n}{\sum_{n=1}^{N} Card\ CO_n} \tag{3}$$

Where N defines the total number of nodes in the network and CO is the content object.

Besides, we measure the cache replacement rate that presents the ratio between the replaced contents and the cached contents for all the nodes in the network.

Finally, we measure the cache hit measured on the network when looking for a content [27]:

$$Cache\ hit = \frac{\sum_{i=1}^{N} hits_i}{\sum_{i=1}^{N} hits_i + \sum_{i=1}^{N} miss_i} \tag{4}$$

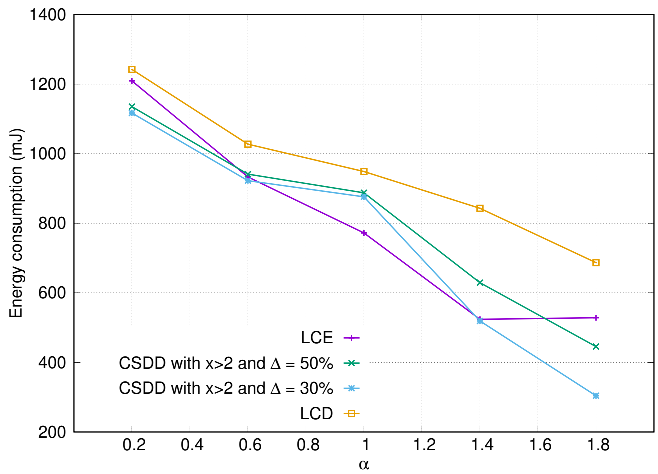
With N the total number of nodes in the network.

5.3. Simulation results

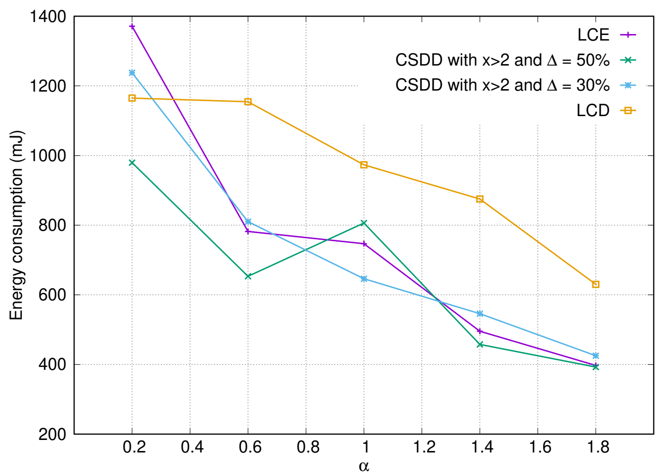
We implement two cache replacement policies: the FIFO and the Popularity-based strategies. Since our interests follow Zipf distribution, we consider that different α describes different scenarios.
As $\alpha \leq 0.6$ describes the low popularity scenarios when there is no rush like students asking for the temperature of a classroom or for information about teachers. This can happen all along the day and by a small number of users at the same time. However, $\alpha \geq 0.6$ presents high popularities scenarios describing rush hours such as 11.30 *am* when many students start sending interests to have an idea about the menu and the queue in front of the restaurant.

5.3.1. Energy consumption

Figure 7 plots the results of energy consumption for the different strategies under different variations of α for two replacement policies: FIFO and Popularity-based.



(a) Energy when using FIFO.



(b) Energy when using Popularity-based.

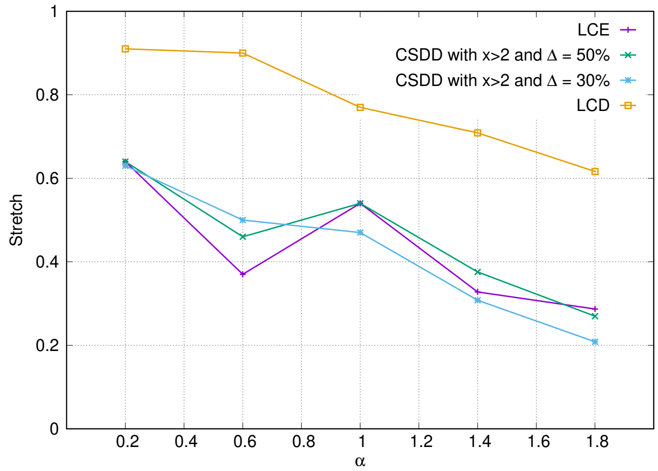
Figure 7: Energy for all the strategies when using LCE and Popularity-based replacement policies.

For both replacement policies, it is noticed that when α increases, the energy consumption decreases for all the strategies under both replacement policies. Indeed, when α increases, some contents significantly overstep others in terms of popularity. Hence, they will be more requested by users and they will be cached more in intermediate nodes. Consequently, most of the requests cross a shorter path. They are not required to reach source nodes to get the corresponding contents. Then, energy consumption decreases. In low popularity scenarios, all the strategies consume more energy. This is due to the fact that the contents are almost requested with the same rate. Then, the replacement happens frequently increasing energy consumption.
It is also worth noting that under the FIFO replacement policy, all the strategies consume more energy. This is explained by the fact that since the interests follow the Zipf distribution in both cases when using FIFO, the probability of replacement is the same for all the contents. Nevertheless, when using the Popularity-based policy, popular contents are kept in the caches. Therefore, the network consumes less energy consumption since the requested contents will be in the cache for a longer time. For instance, for $\alpha = 0.4$, under FIFO, LCE consumes about 1230 mJ and CSDD ($x > 2$ and $\Delta = 30\%$) consumes slightly more than 950 mJ. Yet, under Popularity-based policy, CSDD ($x > 2$ and $\Delta = 30\%$) dissipates just 890 mJ and LCE consumes 1050 mJ.
For FIFO policy, when the popularity of contents is high, LCE acts better than all the other strategies in terms of energy consumption since it caches popular contents everywhere. However, when the Popularity-based policy is used, for high α, CSDD with a x (node degree) higher than 2 and a Δ equals to 50% outperforms LCE. Concerning LCD, we observe that it consumes more energy than LCE and CSDD with $x > 2$ and $\Delta = 30\%$ or 50% . Indeed, LCD just caches contents one hop from the source. Then, when an interest arrives on the network, it has to go to one hop from the source to get the corresponding content.
It is also interesting to mention that in our strategies, only nodes with a degree $x >= 2$ caches contents. Hence, the number of nodes that caches the contents and communication computation decreased. Thus, caching energy is saved. However, sometimes more forwarding energy is dissipated when the number of nodes having this degree is not great.

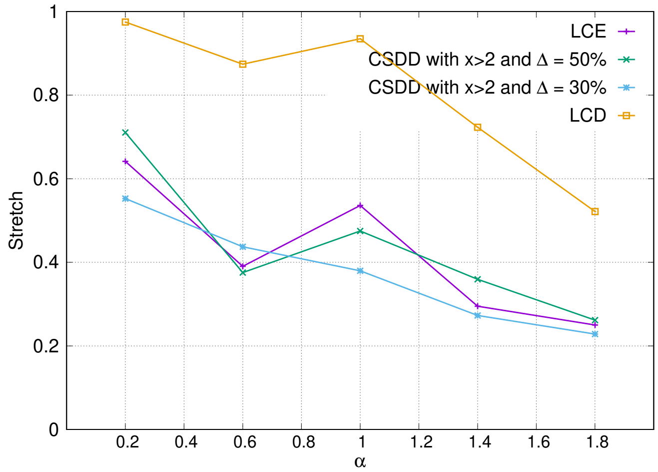
5.3.2. Stretch

Fig 8 represents the results for stretch for LCE, LCD, for our strategy under different variation of Δ and x.

As shown, when FIFO policy is used, in low popularity context (α= 0, 4), CSDD with $x > 2$ and a $\Delta = 30\%$ or $\Delta = 50\%$ outperforms all the other strategies. This is due to the fact, that the content is cached in nodes near to the user and the number of nodes having a degree larger than 2 represents more than 60% of network nodes. Then, crossed distance becomes smaller and probability to cache in nodes with a $x > 2$ is large.
Findings for LCD shows that it has the biggest stretch for both policies. Indeed, since it caches one hop from the source node, every time, the interests have to cross almost all the path to recovering the content.

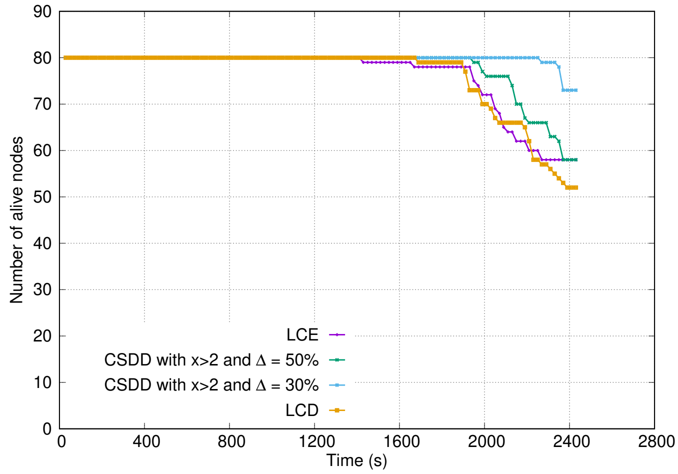


(a) Stretch when using FIFO.

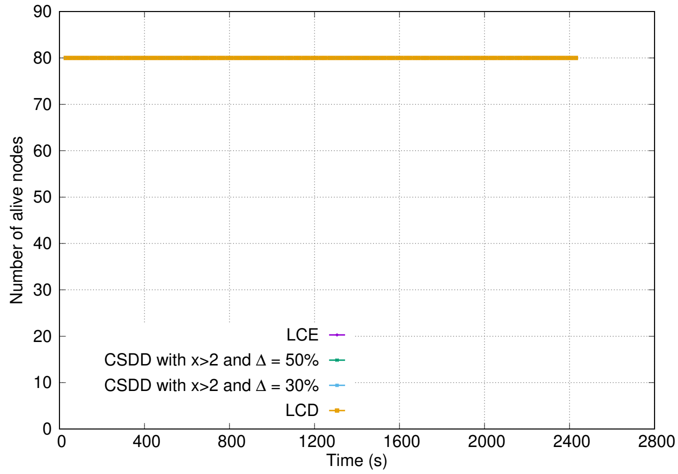


(b) Stretch when using Popularity-based.

Figure 8: Stretch for all the strategies when using LCE and Popularity-based replacement policies.



(a) Network lifetime for low popularity scenarios.



(b) Network lifetime for high popularity scenarios.

Figure 9: Network lifetime for all the strategies when using FIFO under low and high popularity scenarios.

Furthermore, when the popularity of certain contents increases (α increases), the stretch decreases. Certainly, this happens because popular contents will be more available in intermediate nodes.

5.3.3. Network lifetime

FIFO replacement policy. We illustrate in Fig. 9 the network lifetime for different α under the FIFO replacement policy.

As shown, LCD presents the worst results since it always caches in a few particular nodes. Therefore, the node battery expires too soon.

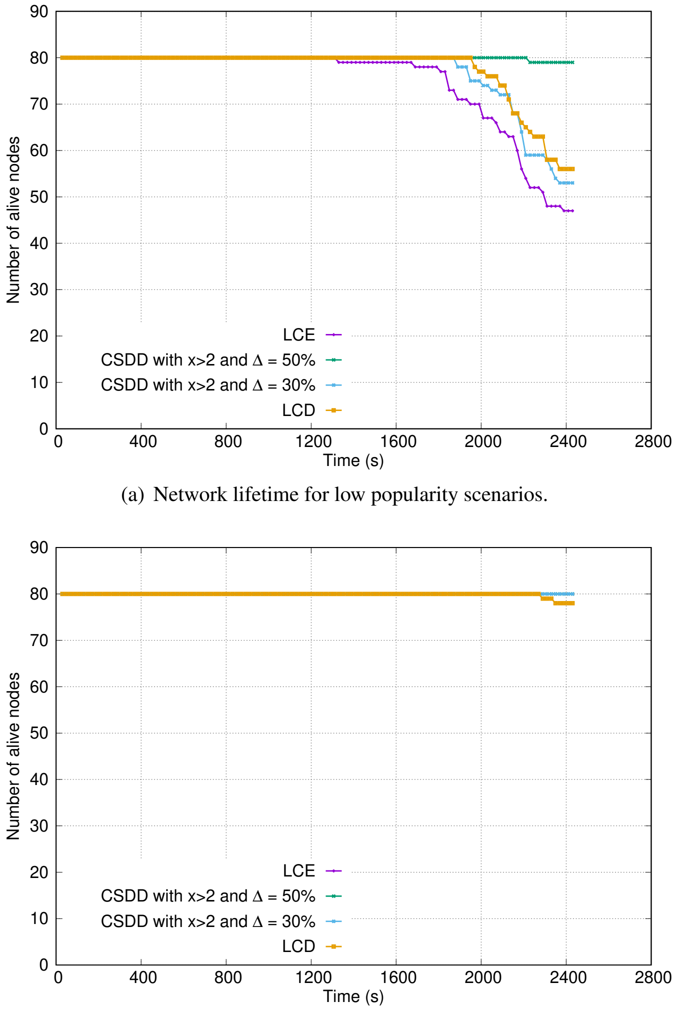
In the context of low popularity ($\alpha < 0.6$), CSDD with $x > 2$ and a $\Delta = 30\%$ or $\Delta = 50\%$ and even with $x > 3$ outperforms all the other strategies even LCE. This is explained by the fact that in CSDD, fewer nodes than in LCE caches the content, all contents have almost the same popularity and the cache size is limited to 6 for all the strategies. Therefore, LCE which caches everywhere realizes replacement frequently and exhausts quickly node battery. For instance, with LCE, the first node die at $t = 1400$ s.

In contrast, in the case of high popularity as depicted in Fig. 9(b), the performance of LCE becomes better but it still does not overcome CSDD with $x > 2$ and a $\Delta = 30\%$ or $\Delta = 50\%$. The battery depletion of the first node happens at $t = 1600$ s in this case. Then, when the popularity increases, the network lifetime increase since popular contents will be more available on intermediate nodes when they are requested.

Popularity-based replacement policy:. Fig. 10 details the findings for network lifetime for different α when using the Popularity-based replacement policy.

As depicted in the figures, the network lifetime is enhanced when applying this type of policy. Comparing to the results shown in the previous Fig. 9, the proposed policy enhances the results when the popularity is high.

On the other hand, in the context of low popularity, CSDD with $x > 2$ and a $\Delta = 30\%$ or $\Delta = 50\%$ still overcomes all the other strategies even LCE. But, the results are quite the same, for ex-



(a) Network lifetime for low popularity scenarios.

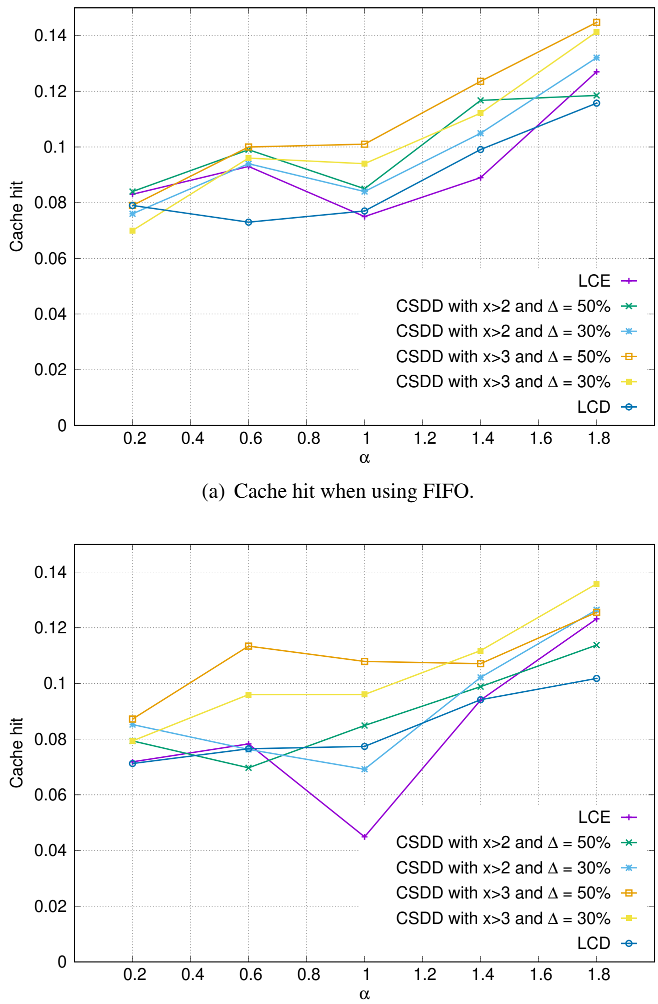
(b) Network lifetime for high popularity scenarios.

Figure 10: Network lifetime for all the strategies when using Popularity based under low and high popularity scenarios.

ample for $\alpha = 0.2$, the first node dies in LCE at $t = 1450$ s. Because with low popularity scenarios, contents almost have the same popularity and the replacement rate does not impact a lot. However, when the content popularity increases, LCE behaves like the proposed strategy (with $x > 2$ and a $\Delta = 30\%$ or $\Delta = 50\%$) and we do not notice any node exhaustion during the duration of the simulation. This is because interests follow the Zipf distribution and the applied replacement policy is Popularity-based. Consequently, replacement happens less frequently and even so interests do not cross the whole path till the source. Then, node energy is saved since the content is available in intermediate nodes.
It is also worth noting that when nodes start to be out of battery, the energy consumption increases, since off nodes will not be able to ensure their functionalities, then the interests cross longer paths to recover the contents.
The LCD network lifetime is extended when using the Popularity-based but it is worse than LCE network lifetime.

5.3.4. Cache hit ratio

We also studied the cache hit ratio for all the strategies under different popularity distribution when using both FIFO and Popularity-based replacement policies.



(a) Cache hit when using FIFO.

(b) Cache hit when using Popularity-based.

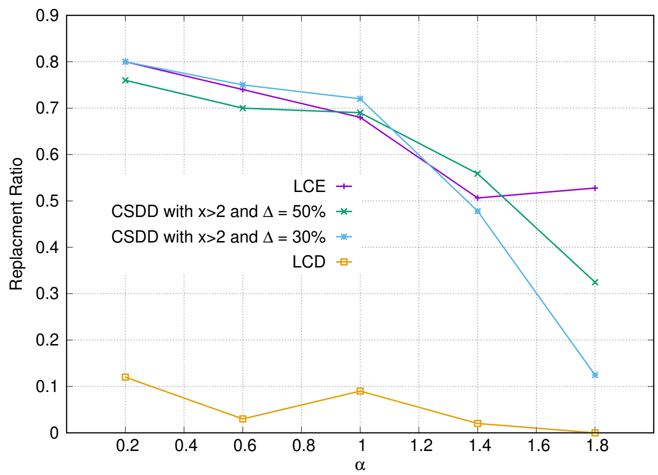
Figure 11: Cache hit for all the strategies when using FIFO and Popularity-based replacement policies.

The results plotted in Fig 11 have shown that when the popularity increases, the cache hit increases when using FIFO and Popularity-based. Indeed, when the α increases, the popularity of certain contents increases and they will be more requested hence they will be more available on the network node.
In Fig. 11(a), for $\alpha = 0.2$ when FIFO is used, we observe that LCE outperforms almost all the strategies (except CSDD with $x > 2$ and a $\Delta = 30\%$ or $\Delta = 50\%$). This is due to the uniform distribution of interest and the maximum caching in nodes. However, in high popularity, CSDD with $x > 3$ and a $\Delta = 30\%$ or $\Delta = 50\%$ reports better performance than LCE.
For Popularity-based replacement policy, for low popularity, this time LCE presents the worst result because it is caching everywhere the same contents. However, in high popularity, it is achieving better results since during the rush hours, there are

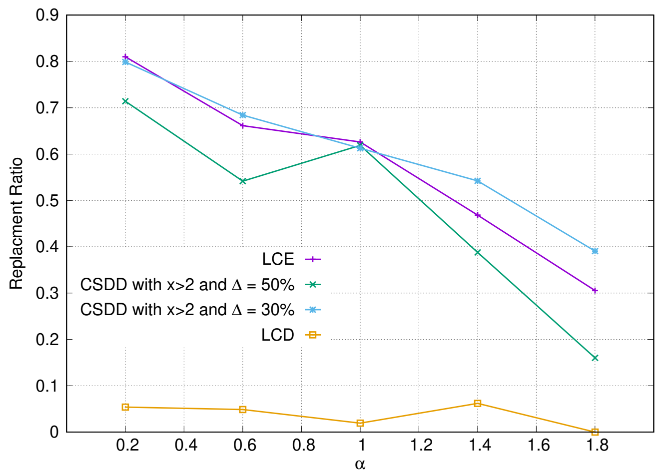
a lot of interests sent on the network requesting the most popular contents. So, it is better to make them available. However, LCE still does not outperform CSDD with a $x > 2$ or $x > 3$ in this case. Therefore, we notice that the values of the caching are low since, in a wireless network, the interests are widely diffused increasing the number of miss since the requested content is not cached in all the network nodes. Then, they will be recording cache miss and decreasing the cache hit ratio.

5.3.5. Replacement rate

In Figure 12, findings for the replacement rate ratio are plotted for all the implemented and proposed strategies when using FIFO and Popularity-based replacement policies.



(a) Replacement rate when using FIFO.



(b) Replacement rate when using Popularity-based.

Figure 12: Replacement rate for all the strategies when using FIFO and Popularity-based replacement policies.

As shown, when the popularity increases, the replacement rate for both policies decreases. In fact, when α is high, just popular contents are requested. For instance in this simulation, content $N1$ is requested with a probability equal to 0.58 for $\alpha = 1.8$. Then, content $N1$ is available in intermediate nodes and its replacement rate is small.

For FIFO policy, when the popularity is low, CSDD with $x > 2$ and a $\Delta = 30\%$ or $\Delta = 50\%$ realizes better results than LCE. However, in case of high popularity, LCE becomes better because popular contents will be cached everywhere and replacement operation will not be frequent. As depicted in the figure, the values of the replacement rate are better under the Popularity-based replacement policy because of its capacity to replace with the less popular content.
It is noted also that LCD, in case of low or high popularity for both FIFO and Popularity-based replacement policies, outperforms all the implemented strategies even LCE. As mentioned before, LCD caches only in one hop from source nodes and then replacement only occurs in these nodes. Then, the overall replacement rate is low.

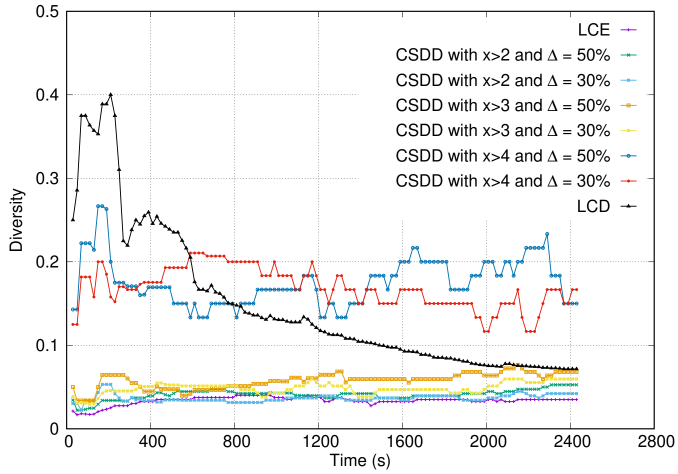
5.3.6. Diversity

Finally, we investigate the diversity for different $\alpha \in \{0.2, 0.6, 1, 1.4, 1.8\}$ for all the strategies when using FIFO and Popularity-based replacement policies. Indeed, strategies that ensure high diversity, may satisfy interests requesting contents with low popularity.

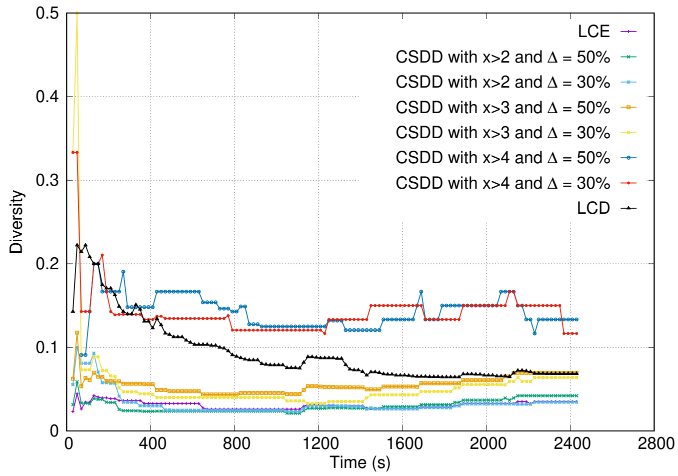
FIFO replacement policy:. Results for FIFO are plotted in Fig 13.

It is observed that when α increases, the cache diversity decreases. Indeed, in the case of high popularity, as mentioned earlier, specific content is more requested and cached. Besides, FIFO ejects from cache contents with the same probability decreasing also the diversity.
In the case of low popularity, we note that LCD present better results than the other strategies at the beginning of the simulation since it caches only on particular nodes which are one hop from the user. At $t = 1200\ ms$, CSDD with $x > 4$ and a $\Delta = 30\%$ or $\Delta = 50\%$ shows better results until the end of the simulation with a diversity larger than 0.2. Absolutely, this is because this strategy hides on a set of very limited nodes. Then, it caches less than the other strategies which increase diversity. In high popularity, as depicted in Fig .13, the diversity slightly decreases since interest distribution (asking for popular content) provides only popular content in caches.

Popularity-based replacement policy:. Results for Popularity-based are shown in Fig 14. An interesting observation comes from Fig 14, the application of the Popularity-based replacement policy decreases the cache diversity. Indeed, this replacement policy enables the caching of the same popular contents. The difference is slightly observed in the case of low popularity but it is obvious in high popularity.
For both content popularity scenarios, LCE shows the worst results in term of cache diversity. The diversity expresses the ratio between the number of unique contents stored in all caches and the total number of contents in the caches. Then, when the number of contents in the caches increases, diversity decreases. This explains the results achieved by LCE. This also explains why CSDD under different x and Δ realizes better results.
For high popularity, LCD outperforms all the strategies and achieves a diversity of 0.5. This is because it caches just one

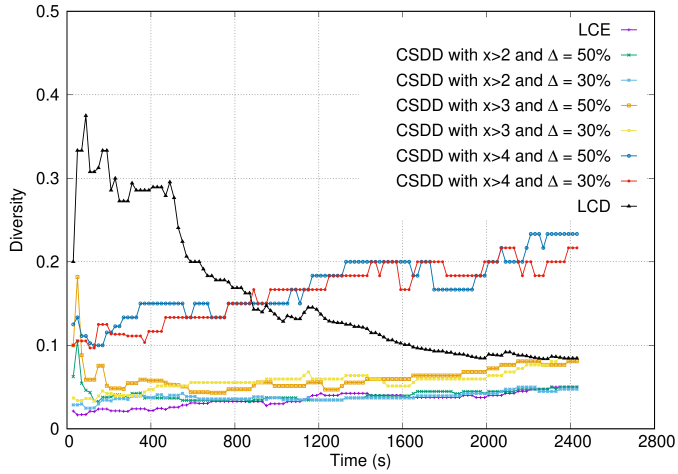


(a) Diversity for low popularity scenarios.

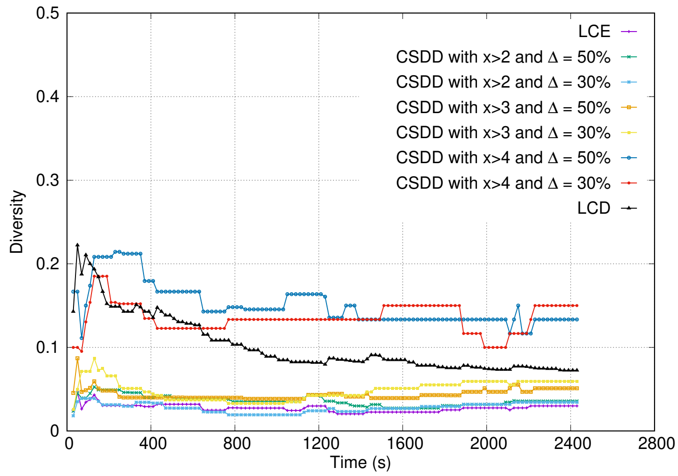


(b) Diversity for high popularity scenarios.

Figure 13: Diversity for all the strategies when using FIFO under low and high popularity scenarios.



(a) Diversity for low popularity scenarios.



(b) Diversity for high popularity scenarios.

Figure 14: Diversity for all the strategies when using Popularity based under low and high popularity scenarios.

hop from the source node the content which decreases the total of cached contents. Besides, since we have different users asking for different contents, diversity is ensured all over the network.

5.4. Discussion

The simulation results showcased the results of CSDD under the variation of several parameters (replacement policy and α) and compared them to LCE and LCD. The proposed caching strategy is based on two parameters, x the node degree and Δ its distance from the source.

For energy consumption, the results highlighted that CSDD with $x > 2$ and $\Delta = 30\%$ or $\Delta = 50\%$ achieves energy saving under the two replacement policies compared to the other strategies. This is due to the fact that the percentage of nodes having this degree is 60% and minimizing caching saves energy. However, CSDD with a $x > 4$ presents the worst results since the percentage of these nodes is just 4% then the content is not cached a lot on the network which increases the energy used while forwarding the data.

As for the network lifetime, CSDD with $x > 2$ and $\Delta = 30\%$ or $\Delta = 50\%$ shows the best results for both replacement policies and under different values of α. Although LCE shows good results in terms of energy, it realizes a bad network lifetime since nodes start dying early. Yet, for a high α, it seems to have better results.

The results have also shown that CSDD can compete LCE in terms of cache hit. In other words, while using the Popularity-based replacement policy and under low popularities, CSDD with different x and Δ shows a better cache hit ratio. In high popularities, CSDD with $x > 2$ and $\Delta = 30$ or $\Delta = 50$ outperforms LCE.

As for the replacement rate, CSDD with and $\Delta = 30$ or $\Delta = 50$ and LCD outperforms LCE. This lay the groundwork for caching in all the node in LCE and then realizing replacement in all the nodes path if caches are full.

Our analysis for diversity also showed that by applying CSDD, better diversity is achieved. In high popularity scenarios, diversity decreases since just popular contents are requested. When FIFO is used, results for diversity are better because the Popularity-based evicts the less popular contents.

We argue that in our strategy caching on more nodes of the path (moving away from the source with = 30%) increases the hit cache and caching on fewer nodes (moving away from the source with = 50%) reduces the replacement rate and energy consumption. Besides, we contend thatcaching on nodes with a degree $x > 2$ achieves a gain in terms of energy, maximizes the network lifetime and decreases the replacement rate.

Finally, we argue that the choice of the degree must be coherent with the percentage of nodes having this degree. For instance, the results showed that the degree $x > 4$ realized bad results since it just represents 4% of the network nodes. Then, the number of potential candidates on which caching is realized, is low. We also contend that our strategy can be enhanced by considering user mobility. In fact, in CSDD, the mobility may balance the traffic load by changing the nodes on which to cache which will maximize the network lifetime. The replacement rate will also decrease and the cache diversity will increase.

6. Conclusion

Cooperative caching can play a major role in handling effectively the queries and in overcoming the situations of none availability of data. Therefore, it reduces the requirement of wireless bandwidth, storage, and energy. In-network caching is one of the best features offered by the content-centric paradigm and one of the most characteristics that motivated us to enable CCN in WSNs. CCN was introduced with LCE which caches contents everywhere realizing a certain degree of redundancy. In this article, we proposed a new caching strategy that aims to find an optimal way to cache the content in order to realize better network performance. For this, we started by presenting different existing caching strategies in the literature that we implemented further. After that, we proposed CSDD an on-path caching strategy in content-centric enabled wireless sensor networks with two parameters to vary (node degree and its distance from the source node). We also proposed a mechanism to detect and overcome the data loop problem caused by the broadcast. Finally, we presented the simulation results in order to show the impact of CSDD on energy consumption, cache hit, replacement rate and diversity. We also realized a comparative study with LCE and LCD. The results showed that CSDD can outperform LCE and LCD when the degree x detains a subset of nodes with an important size. As future works, we will study the impact of the mobility on the proposed strategy and we will enhance our strategy by integrating other parameters.

References

[1] Hashim A. Hashim, B.O. Ayinde, and M.A. Abido. Optimal placement of relay nodes in wireless sensor network using artificial bee colony algorithm. *Journal of Network and Computer Applications*, 64:239 – 248, 2016.

[2] Shuai Gao, Hongke Zhang, and Beichuan Zhang. Energy efficient interest forwarding in ndn-based wireless sensor networks. *Mobile Information Systems*, 2016.

[3] G. Jaber, R. Kacimi, and T. Gayraud. Data freshness aware content-centric networking in wsns. In *2017 Wireless Days*, pages 238–240, March 2017.

[4] Marica Amadeo, Claudia Campo, Antonella Molinaro, and Nathalie Mitton. Named Data Networking: a Natural Design for Data Collection in Wireless Sensor Networks. In *IEEE Wireless Days (WD)*, Valencia, Spain, November 2013.

[5] S. Vural, P. Navaratnam, N. Wang, C. Wang, L. Dong, and R. Tafazolli. In-network caching of internet-of-things data. In *2014 IEEE International Conference on Communications (ICC)*, pages 3185–3190, June 2014.

[6] Melchizedek Alipio, Nestor Michael Tiglao, Antonio Grilo, Fawaz Bokhari, Umair Chaudhry, and Shavez Qureshi. Cache-based transport protocols in wireless sensor networks: A survey and future directions. *Journal of Network and Computer Applications*, 88:29–49, 2017.

[7] Boubakr Nour, Kashif Sharif, Fan Li, Sujit Biswas, Hassine Moungla, Mohsen Guizani, and Yu Wang. A survey of internet of things communication using icn: A use case perspective. *Computer Communications*, 142-143:95 – 123, 2019.

[8] I. U. Din, S. Hassan, M. K. Khan, M. Guizani, O. Ghazali, and A. Habbal. Caching in information-centric networking: Strategies, challenges, and future research directions. *IEEE Communications Surveys Tutorials*, 20(2):1443–1474, Secondquarter 2018.

[9] Xiong Wang, Jing Ren, Tong Tong, Rui Dai, Shizhong Xu, and Sheng Wang. Towards efficient and lightweight collaborative in-network caching for content centric networks. In *Global Communications Conference (GLOBECOM), 2016 IEEE*, pages 1–7. IEEE, 2016.

[10] A. Seetharam. On caching and routing in information-centric networks. *IEEE Communications Magazine*, 56(3):204–209, March 2018.

[11] Maroua Meddeb, Amine Dhraief, Abdelfettah Belghith, Thierry Monteil, Khalil Drira, and Saad AlAhmadi. Cache freshness in named data networking for the internet of things. *The Computer Journal*, 2018.

[12] Z. Zhang, C. Lung, I. Lambadaris, and M. St-Hilaire. Iot data lifetime-based cooperative caching scheme for icn-iot networks. In *2018 IEEE International Conference on Communications (ICC)*, pages 1–7, May 2018.

[13] B. Nour, K. Sharif, F. Li, H. Moungla, A. E. Kamal, and H. Afifi. Ncp: A near icn cache placement scheme for iot-based traffic class. In *2018 IEEE Global Communications Conference (GLOBECOM)*, pages 1–6, Dec 2018.

[14] A. Suki, M. ArifSuhaidi, and HassanSuhaidi HassanIbrahim Abdullahi-Ibrahim Abdullahi. Cache replacement positions in information-centric network. In *International Conference on Internet Applications, Protocol and Services (NETAPPS2015)*, Oct. 2015.

[15] Lorenzo Saino. *On the Design of Efficient Caching Systems*. PhD thesis, University College London, 2015.

[16] Gayathri Tilak Singh and Fadi M. Al-Turjman. A data delivery framework for cognitive information-centric sensor networks in smart outdoor monitoring. *Computer Communications*, 74:38 – 51, 2016. Current and Future Architectures, Protocols, and Services for the Internet of Things.

[17] Bo Chen, Liang Liu, Zhao Zhang, Wenbo Yang, and Huadong Ma. Brr-cvr: A collaborative caching strategy for information-centric wireless sensor networks. In *Mobile Ad-Hoc and Sensor Networks (MSN), 2016 12th International Conference on*, pages 31–38. IEEE, 2016.

[18] neOCampus project. http://www.irit/neocampus.fr. Online; acessed 21 June 2018.

[19] Boubakr Nour, Kashif Sharif, Fan Li, Hassine Moungla, and Yang Liu. A unified hybrid information-centric naming scheme for iot applications. *Computer Communications*, 150:103 – 114, 2020.

[20] S Sundar, V Balakrishnan, R Kumar, and Harish M Kittur. Shortest path solution to wireless sensor networks using edge-based three-point steiner tree concept. In *Wireless Communication Networks and Internet of Things*, pages 51–67. Springer, 2019.

[21] Junaid A. Khan, Cedric Westphal, and Yacine Ghamri-Doudane. A Popularity-aware Centrality Metric for Content Placement in Information Centric Networks. In *ICNC 2018 - International Conference on Computing, Networking and Communication*, Maui, Hawaii, United States, March 2018.

[22] Piyush Charan, Tahsin Usmani, Rajeev Paulus, and Syed Hasan Saeed. A cooperative cache management scheme for ieee802. 15.4 based wire-

less sensor networks. *International Journal of Electrical and Computer Engineering*, 8(3):1701, 2018.

[23] Bilel Saadallah, Abdelkader Lahmadi, and Olivier Festor. *CCNx for Contiki: implementation details*. PhD thesis, INRIA, 2012.

[24] B. Nour, H. Ibn Khedher, H. Moungla, H. Afifi, F. Li, K. Sharif, H. Khelifi, and M. Guizani. Internet of things mobility over information-centric/named-data networking. *IEEE Internet Computing*, pages 1–1, 2019.

[25] Seyed Kaveh Fayazbakhsh, Yin Lin, Amin Tootoonchian, Ali Ghodsi, Teemu Koponen, Bruce Maggs, K.C. Ng, Vyas Sekar, and Scott Shenker. Less pain, most of the gain: Incrementally deployable icn. In *Proceedings of the ACM SIGCOMM 2013 Conference on SIGCOMM*, SIGCOMM '13, pages 147–158, New York, NY, USA, 2013. ACM.

[26] Ghada Jaber, Rahim Kacimi, Luigi Alfredo Grieco, and Thierry Gayraud. An adaptive duty-cycle mechanism for energy efficient wireless sensor networks, based on information centric networking design. *Wireless Networks*, Sep 2018.

[27] César Bernardini, Thomas Silverston, and Olivier Festor. A comparison of caching strategies for content centric networking. In *Global Communications Conference (GLOBECOM), 2015 IEEE*, pages 1–6. IEEE, 2015.