Collaborative informed gateway selection in large-scale and heterogeneous networks

Khulan Batbayar*†, Emmanouil Dimogerontakis*, Roc Meseguer*, Leandro Navarro*, Ramin Sadre†
* Universitat Politècnica de Catalunya
{batbayar,edimoger,meseguer,leandro}@ac.upc.edu
† Université Catholique de Louvain
ramin.sadre@uclouvain.be

***Abstract*—In wireless community access networks, clients tend to reach the Internet through multiple gateway nodes instead of a single default gateway. The mapping of gateways to clients should take into account the perception of network performance from each client node. Network conditions and traffic load can fluctuate and make repeated client-gateway measurements necessary. However, frequent measurements would result in a high communication overhead as well as high processing overhead in gateways and clients. We propose a lightweight client-side gateway selection algorithm by crowd-sourcing monitoring information from neighbor clients, without requiring explicit topology information or a detailed view of the network, while providing an accurate selection as compared to an ideal omniscient approach. Our collaborative gateway selection algorithm achieves good end-to-end performance, such as low latency perceived at client nodes, and fair distribution of the measurements over the gateway nodes. The number of performance measurements triggered by clients are reduced drastically, from n down to 2 measurements per node in each period. An experimental evaluation of our approach shows more than 80% similarity estimation of the gateway performance in the majority of the considered cases. We propose two variants of the gateway selection algorithm, *collaborative-best* and *collaborative-fair*, which yield near optimal gateway selection while utilizing partial information.**

I. Introduction

Wireless Community Networks (WCN) or Do-It-Yourself access networks are gaining more popularity for self-provision of Internet access in underserved regions [1]. WCN allow community members to build their own local network infrastructure by utilizing over-the-counter equipment with minimal technical expertise, by sharing the cost of network infrastructure and by increasing the use of local services. The connectivity to the Internet is arranged through several gateway nodes among the client nodes in a large-scale wireless network. The main challenge is to fairly distribute the available Internet bandwidth among client nodes while maintaining a good overall performance perception at the same time.

The guifi.net [2] Community Network, the largest in the world, consists of more than 34,000 active nodes, with about 12,000 nodes using 394 active web proxy nodes (January 2018) acting as Internet gateways [3], [4]. End-users manually add several known and typically nearby proxy servers to a web browser extension that switches to the next proxy in the list when the current fails. The choice of proxy in each client is not based on performance, but just based on the failure of the current choice, and thus considered quasi-static. In contrast to that, the overall network structure is heterogeneous and the performance of the service offered by the gateway nodes can fluctuate significantly [5]. During rush hours, the Quality Of Experience (QoE) perceived at the end users varies depending on the choice of the gateway node. In order to achieve better QoE, network nodes need to monitor periodically the service performance they achieve from the gateway nodes.

The majority of the gateway selection algorithms in the literature involve explicit measurement of all the gateway nodes by each client, which is not suitable for a large-scale network setting since the cost of measurement can outweigh the benefits of gateway selection. Even though the overhead of a single measurement request is small, in a resource-constrained, large-scale network, the overall traffic generated by measurement requests from every node becomes a major contributor to network congestion. Additionally, many gateway selection algorithms pursue providing full visibility of the gateways just to select one best suitable gateway for the node. Thus, in the end, a significant amount of measurements and message exchanges is left unused. We argue that a partial view of gateways among different small groups of nodes would be effective in terms of reducing in-network traffic, measurement overhead and balancing the client nodes over the gateways.

The main goal of our proposal is to provide an efficient and effective gateway selection by reducing the gateway monitoring overhead, yet providing recent measurements to the client nodes. We argue that the gateway selection algorithm in large heterogeneous WCN can benefit from performance estimations at a lower cost through collaborative information sharing between network nodes.

The contributions of our paper are the following:

1) We propose a set of algorithms for informed gateway selection that reduces the gateway monitoring overhead by random sampling and distribution of the information within a localized group of client nodes. We found a good balance between explicit measurements and random sampling while keeping the information and decisions on the clients, leaving the gateway nodes unmodified.
2) We deploy our algorithm in an experimental heterogeneous environment and quantify its efficiency and effec-

978-3-903176-15-7 © 2019 IFIP

tiveness and the influence of collaborative performance measurements.

II. Related work

There is a large body of work on the gateway selection problem, not only in Wireless Networks but also due to the rising number of Internet of Things devices that need efficient gateway selection algorithms. Most of these works have some important limitations: despite proposing interesting solutions, they tend to fail in considering heterogeneity, monitoring overhead, and lack practical testing in a real-world environment. We therefore only discuss works directly related to ours.

Concerning **Collaboration**, to achieve a low monitoring overhead, some works use collaborative probing [6]–[8]. In [6], the monitoring modules collect statistics about network usage. This approach relies on a centralized system where the collected information is stored, therefore not suitable for a loosely organized WCN. In [7], the clients collaborate to probe the gateways in an elaborate way and use the results to select one. But it is based on a brute-force approach that is not scalable to large networks. The work presented in [8] is closely related to our work, and also uses collaboration among clients to probe the gateways without using brute-force. It presents a synthetic coordinate system of the whole network nodes using the Vivaldi algorithm [9] to estimate the distance between every node in the network including gateway nodes. Each client only probes one gateway and shares the information on this overlay network in a gossiping way.

Concerning **Grouping**, the topic of sensor clustering in WSN [10]–[12] is related. Clustering is used to route sensed information to sinks, a problem very close to the gateway selection. But our problem has important differences: bidirectional traffic, heterogeneity (e.g. use, amount of information, hardware, protocols), and end users (perception of quality of service). Although these works propose interesting solutions, most of them are not applicable to our scenario due to the differences mentioned above.

Concerning **Metrics**, different metrics have been used to make an efficient gateway selection. In [13], gateway selection is based on the load of the gateway while the authors of [14] use a combination of gateway load, route interference and expected link quality metrics to estimate the gateway performance. Recent work in [15] presents a flexible gateway selection based on not only network-based criteria (e.g. economical features or user preferences) for gateway selection. It is a conceptual proposal based on game theory with a thorough information processing at the client node, therefore difficult to implement and deploy on a large-scale network. The PAWS [6] algorithm redirects node requests over gateways based on the flow demand and residual capacity of the gateway.

As compared to previous studies, our proposal offers a more localized collaboration in order to obtain an accurate performance estimation, with the ability to be deployed in large-scale heterogeneous networks with the assistance of client-side metrics.

III. Design decisions

In this section, we describe the two major design decisions in our proposed collaborative and informed gateway selection. The design decisions are motivated by the following desired properties of the selection procedure:

- **Low monitoring overhead:** Periodic performance monitoring requests should not influence the performance of the gateway nodes or even worse overload the gateway nodes or the access network.
- **Fair gateway selection:** Monitoring requests generated by client nodes should be balanced on gateway nodes in order to provide the best QoS. Therefore, each client node should consider during selection how its gateway choice will affect the other nodes.
- **Handling uncertainty:** The selection algorithm should react timely to the changes of gateway performance including node failure, capacity exhaustion, traffic congestion, and degradation of the quality of the service.
- **Incremental and backward compatible:** It should be possible to implement the selection algorithm in an existing environment without interrupting the normal functioning of the network.

A. Collaborative sensing among close neighbors

In collaborative sensing, nodes share measurement results in order to build a common ground of information while keeping the measurement overhead low. More concretely to the problem of gateway selection in WCN, we want nodes to share their gateway performance measurements in order to reduce the number of measurements needed to make an informed gateway selection. Unfortunately, in large networks such as a WCN or a Wireless Sensor Network, collaborative sensing between *all* nodes is not scalable. Instead, nodes are typically organized into more or less small groups and collaboration happens only inside those groups. Different solutions have been proposed to (self-)organize nodes into groups in order to achieve network-wide objectives such as load balancing, fault tolerance, or energy efficiency [16]–[19].

Obviously, sharing of measurements only makes sense among nodes that perceive the gateway performance similarly. This makes the grouping process harder. For example, a simple topological grouping applied to the network shown in Figure 1a might lead to the two groups of nodes N1–N5 shown in Figure 1b. Depending on the characteristics of the network links, the nodes N1, N2 and N3 might see or not see a similar performance of gateway N5. Algorithms like Synthetic coordinate based algorithm and CoPing (see Section II) perform complex measurements and adjustments to organize the nodes and to compensate for the differences between nodes.

In this paper, we propose a simpler mechanism for collaborative sensing that avoids such adjustments. Instead, we allow each network node to choose close neighbors to collaborate with, resulting in individual groups such as the ones depicted in Figure 1c. The underlying assumption is that nodes that are

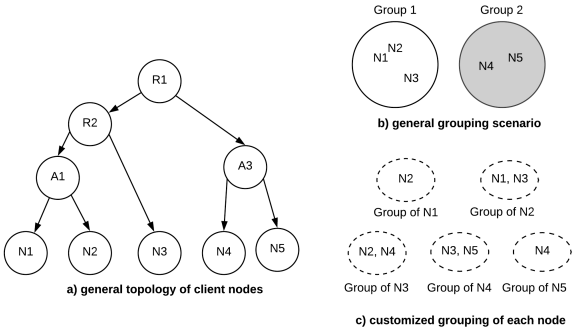


Figure 1: Example scenario for forming a group of collaborating nodes

close to each other (in terms of, e.g., round trip time) probably also share most of the paths to the gateway nodes and therefore they see similar gateway performance.

B. Random sampling

A second fundamental principle in our approach is the use of random sampling. Instead of measuring the performance of all gateways, each node only samples a random subset. When coupled with the collaboration among close neighbors described in Section III-A, a good visibility of the gateway nodes can be still achieved. Existing work suggests that a small number of samples per node can be sufficient. The congestion sensing approach in [20] uses the Power of Two Choices (PoTC) algorithm, a node randomly sense two resources, to distribute the load of the network traffic over the common resources [21], [22]. Similarly, the authors of [23] use the Power of D Choices algorithm to improve the visibility of the load balancing decision in distributed stream processing.

In our work, we perform two gateway measurements per node per measurement round. This small number minimizes the probability that two nodes sense the same gateway. At the same time, it ensures that even an isolated node without any neighbors can choose between at least two gateways, which is important for our gateway selection algorithm (see Section IV-C). Note that a larger number would be also possible, providing a better visibility of the gateway performance at the price of a greater measurement overhead, or even a dynamic adaptation of the number in function of the number of measurements a node has received from its neighbors. However, due to space restrictions, we have restricted the experiments in Section V-C to the fixed number of two gateway measurements per node per round.

IV. Design of the Gateway Selection Algorithms

We propose a client-side informed gateway selection algorithm where network nodes collaborate with closely located neighbor clients to sense the performance of the gateways. Each node keeps a table, called *gateway performance table*, where it stores the results of its own gateway performance measurements for the different gateways as well as the gateway measurement results obtained from its neighbor nodes. The main objective of collaborative sensing is to reduce the in-network traffic and to increase the awareness about gateway nodes at each client node as explained in Section III-A.

Parts of the algorithm can be replaced (for example, we propose two different procedures to select the best gateway). We have therefore structured our approach into different components that are organized in three layers (see Figure 2): The bottom layer provides the performance sensing, that is the measurement of the gateway performance and the identification of close neighbors to collaborate with; the middle layer is in charge of the actual collaboration between network nodes, i.e. the exchange of measurement results; the top layer selects one gateway from the table of measured gateways.

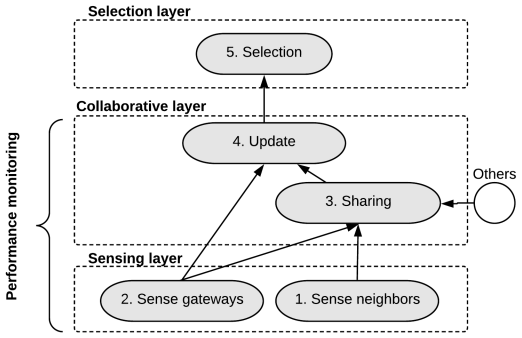


Figure 2: Layered structure of the proposed algorithm

All components of the algorithm run on the client side, i.e. on the network nodes and their execution can be roughly divided into two phases, the *bootstrapping phase* and the *periodic sensing phase*. When a node is activated it starts with an empty gateway performance table. The goal of the bootstrapping phase is, therefore, to identify the set of close neighbors and to receive their measurement results in order to fill the node's table. With this initial version of the table, the node can do a first gateway selection. After this has been done, the node enters the periodic sensing phase where it performs its own measurements (sensing) and exchanges measurement results with neighbor nodes.

The interaction of the different layers during these two phases is depicted in Figure 3. We will now explain the different components in more detail.

A. Sensing layer

1) Component "Sense neighbors": The goal of this component is to sense close neighbor nodes. As explained in Section III-A, the idea is that close neighbor nodes will see similar performances for the different gateways, therefore close neighbors should exchange gateway performance measurements.

There are different possible ways for a node to identify its neighbors. A node can broadcast or multicast discovery messages, passively listen to wireless communication in its

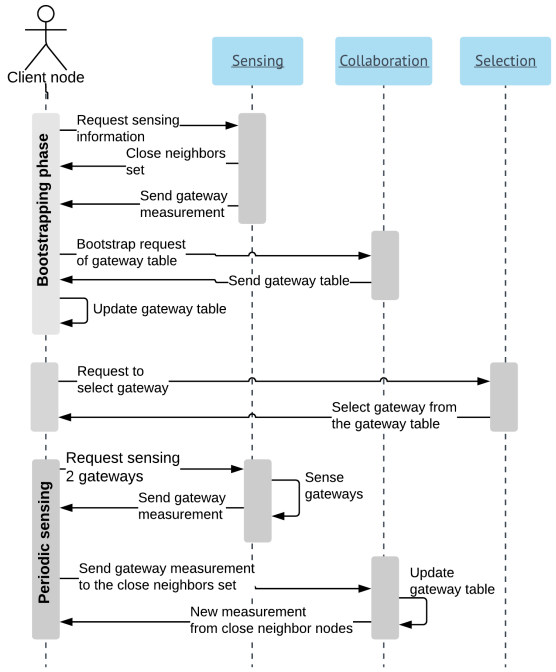


Figure 3: Bootstrapping and periodic sensing phase

neighborhood in promiscuous mode, or use a node discovery service. For example, the guifi.net CN has a central node registry service.

In our implementation of this component, we rely on the registry service. In addition, in order to decide whether a neighbor node is a *close* neighbor, we propose to perform round-trip time (RTT) measurements. We define the RTT as the ping delay between two nodes which is relatively stable in WCN. As explained, WCN can be geographically very extended with multiple hops between nodes and the RTT can be used as a simple metric roughly correlated to the physical or logical distance between two nodes. A node regards neighbors with an RTT below a certain threshold as close neighbors.

The main challenge in this approach is the fine-tuning of the RTT threshold value. As shown later in the experiments in Section V-C2, the smaller the threshold value, the more accurate the gateway performance perception will be since there is a higher probability that the identified close neighbors of a node share the same paths to the different gateways. On the other hand, a larger threshold value leads to a larger set of close neighbors and, therefore, more available measurement results, which can lead to a more informed gateway selection on a node.

2) Component "Sense gateways": As explained in Section III-B, although a node knows all existing gateways, it will not perform performance measurements against all of them. Following the principle of PoTC [20], in each measurement period, a node will randomly pick two gateways and measure their performance. This measurement consists in measuring the time (called "latency" in the following) needed to download a 0.1MB resource via HTTP from a file server located on the Internet. The gateway node performance depends on many aspects including node failure, path congestion, traffic load, the gateway capacity, and service latency. It should be noted that we are not interested in the absolute value of the latency since our goal is only to rank the gateways.

Table I: Example of gateway performance table at each node

Gateway	Latency	Timestamp
Gateway 1	0.6ms	2018-07-01 14:39:14
Gateway 2	0.4ms	2018-07-01 14:37:27

The measurement results will be stored in the node's gateway performance table together with the time of the measurement. This timestamp will be used later in the collaborative and selection layers. Table I shows an example table containing measurement results for two gateways.

As a result, the number of performance measurements between the node and the gateways reduces from a factor of n down to 2 for each node per measurement period, being n the number of available gateways. Two is the minimum number of measurements that an isolated node should make to allow an informed decision. Measuring two random gateways in each measurement period reduces the probability that two neighbor nodes will measure the same gateway.

B. Collaboration layer

Collaboration, i.e. the exchange of gateway measurement results, is limited to close neighbors, assuming that the performance perception of the gateways between close neighbors is similar. The collaboration consists of three parts: (a) Sending own measurement information to close neighbors, (b) receiving information from those nodes, and (c) updating the node's gateway performance table.

Sending: Whenever a node has performed its own performance measurements in the gateway sensing component, the measurement result is directly sent to its close neighbors identified by the neighbor sensing component.

Receiving and updating: Information reception from neighbors is handled by the `Receive` procedure in Algorithm 1. This procedure distinguishes between information received from nodes that are known to be close neighbors of the current node (line 2) and unknown senders (line 13). In the former case, two types of messages are supported: During the bootstrapping phase, a node can request the whole gateway performance table from its closest neighbors in order to quickly obtain an initial version of the table. Such a message is handled in lines 3–5. The second type of messages are messages sent during the periodic sensing phase. They contain performance measurement results from a neighbor node. If a contained measurement result is more recent than the information currently stored in the node's table, the corresponding entry in the table is overwritten (lines 6–11).

When the sender is unknown to the receiving node (lines 14-17), the message is not simply discarded. Instead, the node requests the sensing layer to check whether the sender is a close neighbor. If this is the case, the message is processed.

It should be noted that collaboration between nodes is not transitive, in the sense that a node will not forward a copy of the information received from a neighbor node to its other neighbors. This choice enforces the precision of the estimation within a neighborhood and reduces unnecessary network traffic generated by gossiping.

Algorithm 1 Collaborating with close neighbors

Require: $close\_neighbors = \{C_1, C_2, \ldots, C_k\}$
Require: $gateway\_table$ ▷ Gateway measurement table

```
1: procedure RECEIVE(sender, message)
2:     if sender ∈ close_neighbors then
3:         if message ==' whole table request' then
4:             SEND(gateway_table)
5:         else
6:             for all info ∈ message do
7:                 gw_address ← info.gw_address
8:                 if info.timestamp > gateway_table[gw_address].timestamp then
9:                     gateway_table[gw_address] ← info
10:                end if
11:            end for
12:        end if
13:    else
14:        isNeighbor ← SENSENEIGHBOR(sender)
15:        if isNeighbor then
16:            goto line 3
17:        end if
18:    end if
19: end procedure
20: procedure SEND(message)
21:    for all neighbor ∈ close_neighbors do
22:        SEND_DIRECT(neighbor, message)
23:    end for
24: end procedure
```

C. Selection layer

For the gateway selection, we propose two alternative algorithms which rely on the information stored in the gateway performance table.

The first algorithm is *collaborative-best*. It selects the best gateway, i.e. the gateway with the lowest latency measure, from the gateway performance table within the last measurement period. The algorithm is simple and selfish since it does not consider overall load balancing: Since all nodes in a close neighborhood choose similar gateway performance tables, they will likely select the same gateway with this algorithm. As a possible result, the gateway might get overloaded and performance may degrade for all.

The second selection algorithm is *collaborative-fair* where a node will randomly choose a gateway among the best performing gateway nodes according to some latency threshold, as shown in Algorithm 2. The algorithm builds a list of all gateways for which recent measurements are available and whose measured latency is below the specified threshold (line 4). The procedure is restarted with a doubled threshold if the search does not return at least two gateway candidates (lines 8–10).

We have designed *collaborative-fair* to be conservative in the choice of the gateway. Frequent changes of the gateway could result in sudden short-term performance variations at the gateways, perturbing the performance measurements of the other nodes. Therefore, if the previously selected gateway still belongs to the list of good gateway candidates, the current gateway is not changed (lines 11–12).

Algorithm 2 Collaborative-Fair

Require: $gateway\_table$
Require: $latency_{thr}$ ▷ Threshold for gateway category
Require: $current\_gateway$
Require: $measurement\_period$

```
1: procedure SELECTGATEWAY(latency_thr)
2:     good_gateways = []
3:     for all gw ∈ gateway_table do
4:         if (NOW − gw.timestamp) < measurement_period & gw.latency < latency_limit then
5:             good_gateways ← ADD(gw)
6:         end if
7:     end for
8:     if SIZE(good_gateways)<2 then
9:         return SELECTGATEWAY(2 * latency_thr)
10:    end if
11:    if current_gateway ∈ good_gateways then
12:        return current_gateway
13:    else
14:        return RANDOM(good_gateways)
15:    end if
16: end procedure
```

V. Evaluation

In this section, we evaluate the performance of our approach and compare it with other collaborative and non-collaborative approaches. We begin with an analytic comparison of the different approaches in Section V-A. Then we describe our experimental setup and methodology in Section V-B. The experimental results are presented and discussed in Section V-C.

A. Analytic comparison

The idea behind collaborative sensing is that a node can make an informed gateway selection thanks to gateway performance measurements performed by it and by other nodes in the network. This reduces the amount of measurements the individual nodes have to make. On the other hand, it also introduces a certain overhead due to the messages exchanged between the nodes. Obviously, this overhead is absent in non-collaborative approaches.

The simplest non-collaborative approach is *brute-force* selection: In order to obtain the maximum visibility of the gateway performances, a node has to periodically measure all gateways. Alternatively, the node can decide to only measure *two* gateways, which gives us the *PoTC* [9] approach. Its advantage is the reduction of the number of measurements and if the two gateways are chosen randomly, a distribution of the measurement load over all available gateways. However, the approach only gives a partial visibility of the gateway performance to the node. Finally, a node that could not make any measurements at all would simply select randomly one gateway.

For the collaborative approaches, we consider our approach, sensing based on the synthetic coordinates [8] (referred as Vivaldi based algorithm in the paper), and the CoPing algorithm [7] as they are both informed and collaborative gateway selection algorithms in heterogeneous wireless network. Due to the random selection and the small number (two) of measurements, our approach cannot guarantee a full visibility of the performance of all gateways. On the other hand, its measurement and collaboration overhead is the smallest among the compared collaborative approaches discussed here.

In Vivaldi based algorithm, each node measures a different set of (more than two) gateway nodes, effectively leading to a full visibility of the performance of all gateways. The measurement results are distributed within a set of close neighbors and additional random neighbors. To improve the estimation of the so-called synthetic coordinates between nodes, this distribution is done frequently (n times) within one measurement period.

In CoPing, ancestor nodes are identified that are responsible for measuring the performance of all gateways, thus leading to full visibility. Those ancestor nodes then distribute the performance information down to the descendant nodes. The cost of the collaborative layer of our proposal and the CoPing algorithm is similar, however, the child nodes in the CoPing algorithm need to adjust the information received from the ancestor node according to their RTT between them. The same applies to the Vivaldi based algorithm, which also requires adjustments of the measurement information depending on the sender nodes' synthetic coordinates. As compared to other collaborative algorithms, our collaborative sensing algorithm eliminates this last step, greatly reducing the complexity of our proposal.

Table II summarizes the characteristics of the different approaches.

B. Experimental setup

We have implemented our gateway selection prototype in experimental nodes of a real heterogeneous production network, the guifi.net CN. The nodes are organized into administrative zones which represent the geographical area where the nodes are deployed. Each zone has a different set of gateway nodes where the majority of the client nodes use them to gain access to the Internet. The usual practice for gateway selection in guifi.net CN is *fail-then-connect-next*, where the current selection of the gateway remains until the gateway fails. Access to the Internet is often spare capacity donated by other community members, therefore it has limited bandwidth and capacity.

Our testbed consists of 24 experimental nodes, 12 real, geographically distributed production gateway nodes in the Barcelona zone of guifi.net CN. Our algorithm run periodically every 2 minutes for 1 day period and the traffic and the load of the gateway nodes are real. Experimental nodes introduce 1ms, 3ms, 5ms, and 7ms network delays to mimic the small subset of the network. The CoPing algorithm has not been evaluated in the experiments, as it requires modification in the routing protocol, unfeasible in a production network, even more given the use of more than one.

C. Experimental results

The experiments in this section focus on the precise estimation of the proposed collaborative algorithm as it estimates the performance of the gateway nodes according to its selected close neighbors. We conducted different sensitivity experiments of the algorithm, sensitivity towards RTT threshold, close neighbor nodes size and performance change.

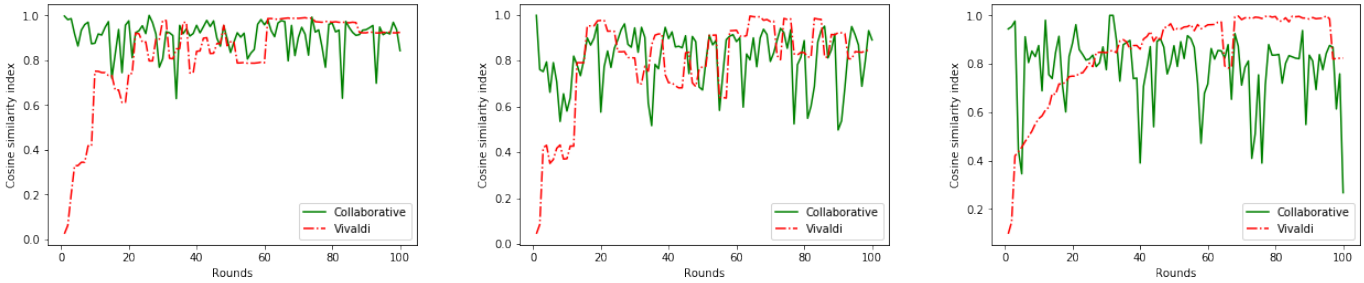
1) Evaluation of the gateway selection: We compared the collaborative and non-collaborative algorithms with our proposed Collaborative-Best and Collaborative-Fair algorithms. Figure 5 shows the Empirical Cumulative Distribution Function (ECDF) of the downloading 0.1 MB file from the selected gateway for each of the studied algorithms. The lower bound is set by the PoTC, where the gateway performance table consists of each node's own two measurements. On the other hand, the upper bound is set by the Brute-Force gateway selection, where the node selects the best gateway at the time of selection out of all gateway nodes performance. As shown by non-collaborative algorithms in Figure 5, both our proposed gateway selection algorithms outperform the worst-case scenario 60% of the time. Moreover, Collaborative-Best gateway selection performs slightly slower (0.1ms slower) than the Brute-Force gateway selection. Among the collaborative algorithms in Figure 5, the Vivaldi based algorithm performs better than our selection algorithms. The reason behind that Brute-Force and Vivaldi based algorithms perform better than our proposed selection algorithm is that they offer full visibility of the gateway nodes at cost of more management overhead (more messages and computation) while our algorithm achieves partial visibility with less management overhead.

2) Sensitivity of the collaborative sensing algorithm: In this experiment, we study the two main factors that influence the precise estimation of the gateway performance, the RTT threshold value used for creating a close neighbors cluster and the number of nodes in the same close neighbors set. We run our experiment for 100 rounds with 3 different RTT threshold value to form a close neighbors set, 5ms, 10ms and 20ms having average 10, 15, 20 close nodes respectively at each node to see the effect of the number of nodes and the threshold values.

In our evaluation, we used the cosine similarity function (0 to 1, 1 being higher similarity), which is the similarity

Table II: Messages cost & gateway visibility at individual nodes per measurement round, g : number of gateways, n: number of close neighbors, r: number of repetitions of the message exchange in one measurement round.

		Message exchange			
		Sensing layer	Collaboration layer	Measurement adjustment	Gateway visibility
Collaborative	**Our proposal**	2	n	No	Partial
	Vivaldi based	> 2	$r \cdot (n + \#random\ nodes)$	Yes	Full
	CoPing	g	$\#descendant\ nodes$	Yes	Full
Non-collaborative	Brute-Force	g	0	N/A	Full
	PoTC	2	0	N/A	2
	Single random	0	0	N/A	1



(a) $RTT < 5ms, cluster = 10$ (b) $RTT < 10ms, cluster = 15$ (c) $RTT < 20ms, cluster = 20$

Figure 4: Sensitivity analysis of RTT threshold value and size of close neighbors cluster

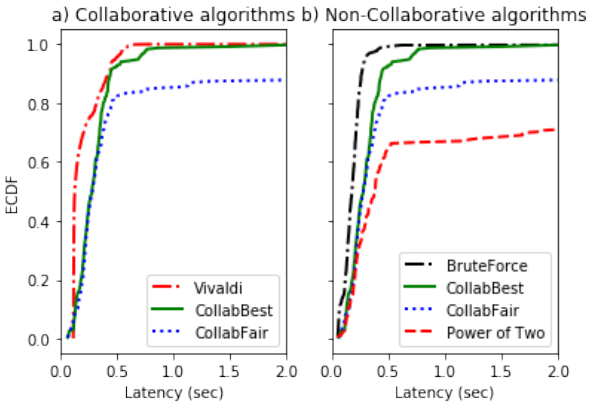


Figure 5: Gateway selection result

between the received measurement against the node's own perception of the gateway performance at the time of receiving information. This experiment conveys how precise estimations our collaborative sensing approach can achieve.

Figure 4 shows the precision of the gateway performance estimation of our collaborative sensing algorithm and Vivaldi based algorithm proposed in [8]. From Figure 4a, both of the collaborative approaches result in near-optimal estimation of the performance of gateway nodes, which is important for the selection of the best suitable gateway for a client node. With Vivaldi based algorithm, the average gateway similarity estimation starts below 0.05 to reach almost 0.98 similarity after 60 rounds, showing a progressive increase of the precision of the estimation as time passes. Comparatively, our collaborative sensing algorithm provides more than 0.8 similarity of estimation throughout the duration of the experiment, as shown in Figure 4a. When increasing the close neighbors RTT threshold value in Figure 4b and Figure 4c, the precision of the gateway performance estimation drops down to average 0.6-0.8. As the RTT threshold value increases, the similarity between nodes in the close neighbors set decreases. The peak values are the result of measurement received from the more closer nodes in the close neighbors set and similarly, lower values are the result of performance measurement received from the distant node in the close neighbors set. On the other hand, the Vivaldi based algorithm outperforms our algorithm in the 10-15th round and in Figure 4c, the precision of estimation stays constantly above 0.8 duration of the experiment.

Our collaborative sensing algorithm's precision of estimation is shown to reach faster precise performance sensing during the initial stage of the algorithm and maintained stable performance throughout the experiment. The Vivaldi based algorithm is stable after 15-20 rounds where it requires thorough (reactive) information exchange between network nodes and gateway nodes and information processing at every node.

Figure 6 shows the effect of the number of nodes in the close neighbors set leaving (decrease) and joining (increase). The experiment runs with the average close neighbors set the size of 8 (RTT<5ms) and 10 gateways. Each measurement round, the number of measured gateways are different depending on what gateways the nodes in the close neighbors set are sampling. In round 20, we removed 2 client nodes from the experiment, however, the size of the measured gateways and precision of estimation is not affected. We further removed 2 clients at the round 32, then the number of measured gateways decreased but the precision of estimation remains the same. We observed the ideal number of the nodes in the close neighbors set should be around $(available\_gateways)/2$ to

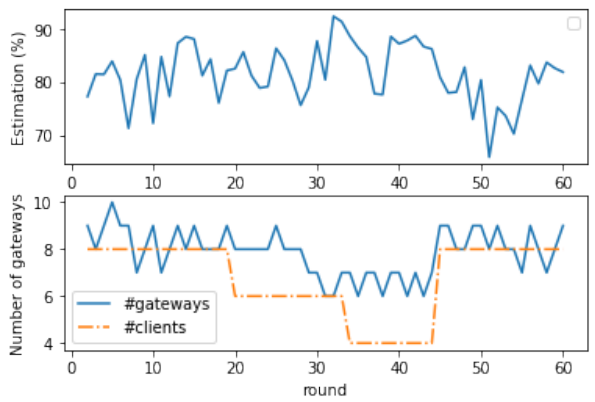


Figure 6: Sensitivity analysis (nodes leaving/joining)

Table III: Average message exchange at individual node per measurement round

		Collaborative layer	
	Sensing layer	Send	Receive
Our algorithm	2	30	30
Vivaldi based algorithm	22	110	156

provide enough gateway measurements.

In Table III, we explore the average number of messages exchanged between Sensing layer and Collaborative layer at individual node per measurement round in the testbed of Figure 4b, with the close neighbors set containing an average 15 close neighbors. For our algorithm, each round the node senses the performance of 2 gateway and sends and receives 2 new gateway measurements from close neighbors. In the meanwhile, Vivaldi based algorithm senses the majority of the gateways and the all the neighbor nodes, then proceed to send whole gateway table entries to the neighbor nodes and receive information from others. Table III, as compared to Vivaldi based algorithm, shows that our algorithm reduces sensing overhead by 10x, message exchange between network nodes by 5x on average.

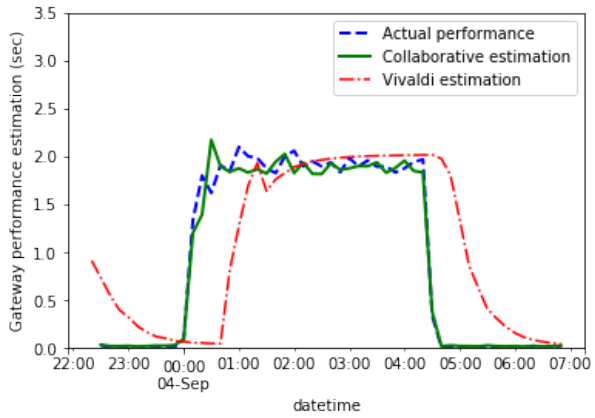


Figure 7: Gateway performance change

3) Self-adaptation of the collaborative sensing: The most common problem of a dynamic network is that it is susceptible to performance changes including the scenario of a short-termed valley on gateway performance and the effect of congestion. Figure 7 shows the performance change adaptation of both cases.

In Figure 7, we introduced a 200ms delay to the selected gateway after the 100th round and removed the delay after 150 rounds. We plotted the estimated gateway performance perceived at our proposed solution (solid line) with the actual measurement of the node (dashed line) and Vivaldi based algorithm estimation (dotted-dash line). The Vivaldi based algorithm took 30 rounds to adjust to the actual gateway performance change, comparatively, our proposed algorithm is able to sense the gateway performance increase after 2 rounds, upon receiving the measurement from the neighbor node in the close neighbors set, and adjust its own gateway table. When eliminating the delay, the Vivaldi based algorithm took 20 round to adjust back to the normal performance change while our algorithm sensed the performance change at the same round.

In our proposed algorithm, the adaptation period to any performance change depends on the probability of receiving the specific gateway performance information from the other nodes, therefore, results in faster convergence within a few rounds. The Vivaldi based algorithm, on the other hand, is slower to adapt to changes, moving towards the direction of the change, therefore not able to timely react to the gateway performance change.

VI. Conclusion and Future Work

This paper addresses the informed gateway selection problem by collaborative performance sensing within close neighbors, drastically reducing the information to share (a reduction factor from n to 2) to achieve good QoE at client nodes and overall fair distribution of the capacity of gateway nodes. Our selection algorithms are simple yet effective, that is infrastructure and technology agnostic and support incremental implementation (compatible with WCN networks that grow organically in the number of clients and number of gateways). Experiments show that the precision estimation of our proposed algorithm is high (>80%) throughout the experiments, which results in high-quality Internet access for clients. By utilizing the partial knowledge of the gateway performance information, the *collaborative-best* and *collaborative-fair* variants perform close to a brute force algorithm.

Future work will explore adaptive sensing to further reduce the monitoring overhead depending on the number of close neighbor nodes, the number of gateway nodes by adjusting the measurement period accordingly. We plan to incorporate fault tolerance and capacity planning of gateway selection in an extended version of the algorithm. Further, we will test the scalability of our proposed algorithm in the real heterogeneous production network of guifi.net, with a large ratio of client nodes versus gateway nodes.

ACKNOWLEDGMENT

This work was supported by the EU Horizon 2020 Framework Program project netCommons (H2020-688768), by the EMJD-DC program, by the Spanish Government under contract TIN2016-77836-C2-2-R, and by the Generalitat de Catalunya as Consolidated Research Group 2014-SGR-881.

REFERENCES

[1] C. Rey-Moreno, W. Tucker, N. Bidwell, R. , S. , and S.-R. , "Experiences, challenges and lessons from rolling out a rural wifi mesh network," 01 2013.

[2] R. Baig, R. Roca, F. Freitag, and L. Navarro, "guifi.net, a crowdsourced network infrastructure held in common," *Computer Networks*, vol. 90, pp. 150–165, 2015.

[3] D. Vega, R. Baig, L. Cerdà-Alabern, E. Medina, R. Meseguer, and L. Navarro, "A technological overview of the guifi.net community network," *Computer Networks*, vol. 93, pp. 260 – 278, 2015.

[4] M. Selimi, A. M. Khan, E. Dimogerontakis, F. Freitag, and R. P. Centelles, "Cloud services in the guifi.net community network," *Comput. Netw.*, vol. 93, no. P2, pp. 373–388, Dec. 2015. [Online]. Available: http://dx.doi.org/10.1016/j.comnet.2015.09.007

[5] E. Dimogerontakis, R. Meseguer, and L. Navarro, "Internet access for all: Assessing a crowdsourced web proxy service in a community network," in *International Conference on Passive and Active Network Measurement (PAM)*. Springer, 2017, pp. 72–84.

[6] A. Abujoda, D. Dietrich, P. Papadimitriou, and A. Sathiaseelan, "Software-defined wireless mesh networks for internet access sharing," *Computer Networks*, vol. 93, pp. 359–372, 2015.

[7] B. J. Ko, S. Liu, M. Zafer, H. Y. S. Wong, and K.-W. Lee, "Gateway selection in hybrid wireless networks through cooperative probing," in *Integrated Network Management (IM)*. IEEE, 2013, pp. 352–360.

[8] E. Dimogerontakis, J. Neto, R. Meseguer, L. Navarro, and L. Veiga, "Client-side routing-agnostic gateway selection for heterogeneous wireless mesh networks," in *Integrated Network and Service Management (IM)*, 2017, pp. 377–385.

[9] F. Dabek, R. Cox, F. Kaashoek, and R. Morris, "Vivaldi: A decentralized network coordinate system," in *Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications (SIGCOMM)*. ACM, 2004, pp. 15–26.

[10] M. M. Afsar and M. Tayarani-N., "Clustering in sensor networks: A literature survey," *J. Network and Computer Applications*, vol. 46, pp. 198–226, 2014.

[11] N. Sabor, S. Sasaki, M. Abo-Zahhad, and S. M. Ahmed, "A comprehensive survey on hierarchical-based routing protocols for mobile wireless sensor networks: Review, taxonomy, and future directions," *Wireless Communications and Mobile Computing*, vol. 2017, 2017.

[12] A. Abuarqoub, M. Hammoudeh, B. Adebisi, S. Jabbar, A. Bounceur, and H. Al-Bashar, "Dynamic clustering and management of mobile wireless sensor networks," *Computer Networks*, vol. 117, pp. 62–75, 2017.

[13] Y. Kim, Y. Jeong, M. Seo, and J. Ma, "Load-balanced mesh portal selection in wireless mesh network," in *MILCOM 2007 - IEEE Military Communications Conference*, Oct 2007, pp. 1–6.

[14] U. Ashraf, S. Abdellatif, and G. Juanole, "Gateway selection in backbone wireless mesh networks," in *Wireless Communications and Networking Conference, (WCNC)*. IEEE, 2009.

[15] X. Wang, D. Qu, K. Li, H. Cheng, S. K. Das, M. Huang, R. Wang, and S. Chen, "A flexible and generalized framework for access network selection in heterogeneous wireless networks," *Pervasive and Mobile Computing*, vol. 40, pp. 556–576, 2017.

[16] W. B. Heinzelman, A. P. Chandrakasan, and H. Balakrishnan, "An application-specific protocol architecture for wireless microsensor networks," *Trans. Wireless. Comm.*, vol. 1, no. 4, pp. 660–670, Oct. 2002.

[17] C. Nie, H. Wu, and W. Zheng, "Latency and lifetime-aware clustering and routing in wireless sensor networks," in *2016 IEEE 41st Conference on Local Computer Networks (LCN)*, Nov 2016, pp. 164–167.

[18] Y. Fernandess and D. Malkhi, "K-clustering in wireless ad hoc networks," in *Proceedings of the Second ACM International Workshop on Principles of Mobile Computing*, ser. POMC '02. New York, NY, USA: ACM, 2002, pp. 31–37.

[19] Y. Liao, H. Qi, and W. Li, "Load-balanced clustering algorithm with distributed self-organization for wireless sensor networks," *IEEE Sensors Journal*, vol. 13, no. 5, pp. 1498–1506, May 2013.

[20] M. Mitzenmacher, "The power of two choices in randomized load balancing," *IEEE Transactions on Parallel and Distributed Systems*, vol. 12, no. 10, pp. 1094–1104, 10 2001.

[21] H. Zhang, J. Zhang, W. Bai, K. Chen, and M. Chowdhury, "Resilient datacenter load balancing in the wild," in *Special Interest Group on Data Communication (SIGCOMM)*, 2017, pp. 253–266.

[22] S. Ghorbani, B. Godfrey, Y. Ganjali, and A. Firoozshahian, "Micro load balancing in data centers with drill," in *ACM Workshop on Hot Topics in Networks*. ACM, 2015, pp. 17:1–17:7.

[23] M. A. U. Nasir, G. D. F. Morales, N. Kourtellis, and M. Serafini, "When two choices are not enough: Balancing at scale in distributed stream processing," *CoRR*, vol. abs/1510.05714, 2015.