A self-organized backpressure routing scheme for dynamic small cell deployments

José Núñez-Martínez, Jorge Baranda and Josep Mangues-Bafalluy

Centre Tecnològic de Telecomunicacions de Catalunya (CTTC)
Av. Carl Friedrich Gauss, 7
08860 Castelldefels (Barcelona), Spain
e-mail: [jose.nunez,jorge.baranda,josep.mangues]@cttc.cat

Abstract

The increase of demand for mobile data services requires a massive network densification. A cost-effective solution to this problem is to reduce cell size by deploying a low-cost all-wireless Network of Small Cells (NoS). These hyper-dense deployments create a wireless mesh backhaul amongst Small Cells (SCs) to transport control and data plane traffic. The semi-planned nature of SCs can often lead to dynamic wireless mesh backhaul topologies.

This paper presents a self-organized backpressure routing scheme for dynamic SC deployments (BS) that combines queue backlog and geographic information to route traffic in dynamic NoS deployments. BS aims at relieving network congestion, whilst having a low routing stretch (i.e., the ratio of the hop count of the selected paths to that of the shortest path). Evaluation results show that, under uncongested conditions, BS shows similar performance to that of an Idealized Shortest PAth routing protocol (ISPA), while outperforming Greedy Perimeter Stateless Routing (GPSR), a state of the art geographic routing scheme. Under more severe traffic conditions, BS outperforms both GPSR and ISPA in terms of average latency by up to a 85% and 70%, respectively. We conducted ns-3 simulations in a wide range of sparse NoS deployments and workloads to support these performance claims.

Keywords: `dynamic, opportunistic, mobile backhaul, transport, small cell, backpressure, routing`

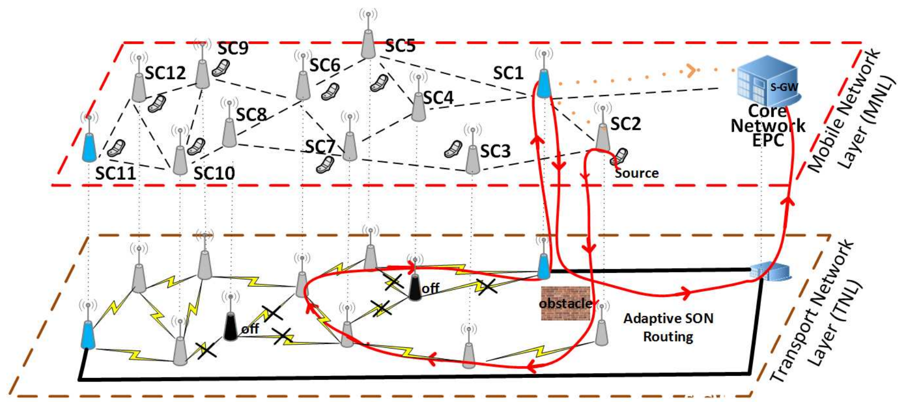

Figure 1: Portion of an all-wireless semi-planed and hyper-dense yet irregular NoS deployment due to obstacles and SCs powered off opportunistically.

1. Introduction

The ever increasing demand for wireless data services has given a starring role to dense small cell (SC) deployments, as increasing frequency re-use by reducing cell size has historically been the most effective and simple way to increase capacity [1]. Such densification, particularly substantial in densely populated areas, entails several challenges, both at the mobile network layer (MNL), specified by 3GPP, and at the underlying transport network layer (TNL). As for the former, the idea of network of small cells (NoS) has been proposed to confine control plane and data plane traffic in the local environment [2]. As for the latter, Figure 1 reveals how SCs, equipped with an additional wireless radio, can create a wireless mesh backhaul to direct control and data plane traffic between them or towards the core network (Evolved Packet Core, or EPC, for LTE networks). The resulting deployment yields improvements in terms of cost, coverage, ease of deployment, and capacity.

However, such deployments imply several challenges at the TNL level. On the one hand, the semi-planned and low-cost nature of the NoS placements will inevitably lead to irregular (or sparse) topologies, SC failures, or disconnection due to obstacles (e.g., wireless link amongst SC_1 and SC_2 in Figure 1), wireless link variability (e.g., due to adverse weather conditions), or vandalism. On the other hand, the wireless backhaul is subject to traffic dynamics. The study in [3] shows that a large fraction of mobile subscribers generate traffic only a few days a week and a few hours during the day. The activation of all SCs when a low fraction of mobile subscribers are using the network results in unnecessary resource consumption and interference. In an hyper-dense SC environment, a possibility is to power off some SCs (e.g., SC4 and SC8 in Figure 1) selectively during low load conditions (e.g., at night), whilst still being able to serve all the traffic. Despite energy efficiency gains, these opportunistic SCs [4] could substantially alter the wireless backhaul topology, hence contributing to dynamic and often sparse deployments.

The dynamicity of the above context may render transport protocols such as MPLS-TP [5], traditionally used in wired TNLs, unsuitable for an all-wireless NoS. A challenge for the TNL is to design a dynamic routing protocol that operates efficiently in large-scale and changing SC topologies, whilst meeting mobile traffic demands. A strategy to tackle large-scale multi-hop wireless topologies is geographic routing [6]. A well known problem of geographic routing is how to react when packets get trapped in a dead end/local minimum (i.e., when there is not any neighboring node closer to the packet destination). In such situations, most geographic routing protocols have their own recovery methods to find a detour path when they reach a local minimum [7]. However, these strategies entail a substantial increase in control overhead as well as an increment of the per-node routing state, required to build alternative routes, hence compromising their scalability in sparse deployments. Further, despite eventually circumventing network voids, geographic routing can lead to network congestion due to a misuse of network resources, and a high routing stretch (i.e., the ratio of the hop count of the selected paths to that of the shortest path) due to the lack of flexibility of the route recovery method.

To address these problems, we present Backpressure routing for dynamic Small cell deployments (BS), a self-organized routing scheme for the TNL that combines queue backlog, geographic information, and information carried in the packet to route traffic in dynamic and often sparse NoS deployments. Our previous work on backpressure routing ([8, 9]) helped us to evaluate the potential of combining geographic and backpressure routing when applied to regular (i.e., grid-like) multi-hop wireless networks. However, practical deployments are far from regular due to the reasons explained above. The main contribution of this paper is to tackle this fundamental aspect by presenting a new scheme that retains the beneficial features of our previous schemes. A preliminary extended abstract of this paper identifying the potential of using backpressure routing for sparse SC deployments appears in [10]. The novelty in this paper resides in the presentation of the resulting scheme and an extensive evaluation in a wide range of scenarios as well. Instead of using a complex and resource consuming geographic recovery method to deal with dead ends, we propose a self-organized, low-overhead, scalable, and decentralized routing approach that makes the most out of the network resources. Unlike geographic routing schemes, the resulting approach neither requires routing recovery methods nor incrementing the per-node state to dynamically adapt to the current wireless backhaul topology.

Extensive ns-3 [11] simulations results validate the robustness of BS under a wide variety of wireless mesh deployments and workloads. Under uncongested traffic demands, BS showed a latency and routing stretch performance close to an idealized single path routing protocol (ISPA), which is aware of the global current network topology without consuming air resources. ISPA is taken as an abstraction of traditional TNL protocols of core networks and wired mobile backhauls (e.g., MPLS-TP [5]). In turn, BS improved the latency results obtained by GPSR [12], taken in general as benchmark for comparison against geographic routing featuring void circumvention mechanisms. In the case of more severe traffic conditions, BS outperforms both GPSR and ISPA showing a reduction in terms of average latency of up to a 85% and 70%, respectively, due to its inherent load balancing capabilities while serving the offered load and maintaining a low routing stretch.

The remainder of this paper is organized as follows. The related work in Section 2 is followed by a list of problems to tackle when using backpressure routing in sparse deployments in section 3. The resulting solution with the details on the operation of BS are provided in Section 4. Finally, Section 5 discusses the simulation results before concluding the paper with Section 6.

2. Related Work

In the following, we describe the main ideas behind backpressure and geographic routing protocols.

2.1. Geographic Routing

Geographic routing [6] represents a stateless and scalable method to route packets in a mesh backhaul using position information. Usually, packets are routed using greedy forwarding. In greedy mode, each node forwards a packet to an immediate neighbor which is geographically closer to the destination node. However, local minima are an important issue for geographic routing under sparse deployments. A packet reaches a local minimum when its distance to the destination is smaller than that of all its neighbors. Most of the work done based on the circumvention of local minima ends up in a routing strategy that implies investing network resources (e.g., air resources) to circumvent the network void [7].

A summary of the protocol used as reference in most of the work based on geographic routing and circumvention of network voids follows. GPSR [12] belongs to the category of position-based routing, and proposes two modes of operation to forward packets: greedy and recovery (or perimeter) mode. The greedy mode is the default mode of operation until a packet reaches a local minimum. GPSR recovers from a local minimum entering in recovery mode, which performs routing operations based on the right-hand rule. As mentioned in [12], this rule states that when a packet arrives at node x from node y, that is, it traverses edge (x,y), the next edge traversed is the next one sequentially counterclock-wise from link (x,y). Nevertheless, the right hand rule can incur into routing loops if the graph is not planar, that is, there are cross-edges in the graph. The packet resumes forwarding in greedy mode when it reaches a node whose distance to the destination is smaller than the distance from the local minimum to the destination.

2.2. Backpressure Routing

The intellectual roots of dynamic backpressure routing for multi-hop wireless networks lies in the seminal work by Tassiulas and Ephremides [13]. Despite this approach promises throughput optimality, two main practical problems were not tackled in this work: i) the high complexity of queue structure and ii) the high end-to-end latencies. Based on [13], several modifications have been proposed to the backpressure algorithm focused on decreasing the complexity of queue structures or decreasing the attained latency. The shadow queue concept in [14] reduces the queue complexity of the original backpressure framework by maintaining a counter per destination instead of a queue per flow. Authors from [15] tackle latency problems associated with the original backpressure algorithm by increasing its number of queues. Their scheme proposes per-hop queues in each node. That is, each node maintains a separate queue per packets that have to be delivered to each destination within a certain number of hops. Even though these proposals alleviate to some extent the original backpressure shortcomings, they still require per-flow or per-destination information.

Closer to our approach, Neely extended the concepts of Tassiulas and defined the Lyapunov-drift-plus-penalty ratio to optimize wireless multihop networks [16]. The theoretical strengths derived from this work have recently increased the interest on practical implementation of the Lyapunov-drift-plus-penalty ratio over wireless networks. Based on Neely's theoretical work, some practical single-queue backpressure routing implementations can be found. In the context of dense sensor networks, authors from [17] consider many-to-one traffic communications, and use a single LIFO queue per node to reduce end-to-end latency.

In our previous work in the context of mesh backhauls ([8, 9]), we consider any-to-any traffic communications. Instead of keeping per-flow or per-destination information, we combine backpressure and geographic information to reduce latency on regular topology deployments with a single FIFO queue per node. Nevertheless, the problem of a void handling procedure for a practically deployable scheme remained unanswered up to our preliminary work in [10]. Next, we present in this paper a detailed description of the resulting algorithm and demonstrate by means of extensive simulations that BS retains the good features of the geographic and backpressure philosophies.

3. Backpressure Routing Challenges under Sparse Deployments

Our scheme uses the theoretical ideas of the Lyapunov drift-plus-penalty method [16], refined by the addition of new techniques needed for tackling practical small cell (SC) deployments. We first define the basic routing algorithm and then we identify the challenges for sparse SC deployments.

3.1. Basic Backpressure Routing Algorithm

The theoretical foundations of our scheme are based on the Lyapunov optimization framework detailed in [16]. Note that this framework can omit the statistics of the random events happening in the network, such as variability of wireless link rates or packet arrival. This eases the design of routing algorithms for dynamic wireless networks. A descriptions of the resulting self-organized backpressure routing policy follows.

Let i and j denote two neighboring SCs/nodes. Our basic scheme, presented for regular (i.e., grid-like) deployments in [8], is based on the calculation of weights for every link (i, j). The weights, denoted as w_{ij}, are defined as follows:

$$w_{ij}(t) \triangleq \Delta Q_{ij}(t) - V_i(t) c^d_{i,j}(t) \tag{1}$$

When taking forwarding decisions, each packet is forwarded to node j^* (amongst all nodes j neighbors of i) with the maximum link weight w_{ij}. That is, the selected neighboring node j^* is such that:

$$j^* \triangleq \arg\max_{j \in J} w_{ij}(t), \tag{2}$$

where J is the set of neighbors of i. In this sense, it is referred to as the Max-weight policy. Equation 1 presents three key components:

Backpressure Routing: The minimization of the Lyapunov drift ($\Delta Q_{ij}(t)$) between neighboring SCs is essential for evenly balancing the traffic load among the wireless links and nodes in the mesh backhaul. Besides, we show in the following section that it allows the eventual circumvention of connectivity voids.

Geographic Routing: The cost function $c^d_{i,j}(t)$, proposed in [8], uses geographic information to penalize forwarding decisions that move the packet away from the destination d ($c^d_{i,j} = 1$), and rewards routing decisions otherwise ($c^d_{i,j} = -1$).

The V parameter: As defined in [9], $V_i(t)$ is a non-negative function in charge of finding the appropriate tradeoff between geographic routing (getting as close as possible to the destination) and backpressure routing (evenly distributing the load among all neighbors), adapting automatically its value to the network state. The value of $V_i(t)$, upper bounded and initially set to the queue size limit (i.e., Q_{MAX}), is calculated as follows:

$$V_i(t) \triangleq Q_{MAX} - \max(Q_j(t)); j \in J \cup \{i\}. \tag{3}$$

Thus, low values of $V_i(t)$ come as a consequence of detecting a node that starts to be congested, and so, a higher weight will be given to move to a lower congestion state by reducing the drift. Contrarily, in uncongested networks, the path with the shortest distance will be traversed by the packet.

3.2. Limitations of Basic Backpressure in Sparse Deployments

Despite the potential of the above framework, a few problems remain to be solved. Here, using network simulation with ns-3 [11], we analyze how our scheme reacts under sparse deployments. Figure 2 depicts a 5x5 grid mesh backhaul, and assume that a percentage of the SCs (shaded nodes) have been powered off at a certain instant. Each SC maintains a single FIFO queue with Q_{MAX} equal to 200 packets. We assume that for each SC, horizontal and vertical neighbors are 1-hop neighbors whereas diagonal neighbors are considered 2-hop

Figure 2: Sparse wireless backhaul scenario

neighbors. Within this scenario setup, SC_6 sends a 2Mbps UDP CBR flow to SC_8. Thus, given that the SC_7 is unavailable, the shortest path under this mesh backhaul configuration has 4-hops through SCs (11, 12, 13, and 8) and (1, 2, 3, and 8).

Figure 3 plots the time evolution of the queue backlog in SC_6, the one facing a dead end due to SC_7 being switched off at time $t = 5s$. The routing scheme is configured with different fixed V values and using the variable-V algorithm described above. Interestingly, Figure 3 reveals that packets remain trapped up to a certain extent. *The first aspect to point out is that such a scheme requires to fill the queue of the SC being the local minimum up (i.e. SC_7) to a limit in which routing decisions emphasize more the reduction of queue backlog differentials rather than geographic proximity to the destination. Nonetheless, packets remain trapped in SC_6 once the queue backlog is below this limit (i.e., at instant $t = 35s$ when the flow terminates).*

We observed how different configurations of the V parameter yield different queue backlogs to enable the use of queue backlog differentials, hence allowing packet to escape from the network void. With the decrease of the V parameter, the queue threshold required to start taking routing decisions based on queue backlog differentials also decreases, hence causing a decrease of queueing latencies. *The second point to remark is that the configuration of the V parameter is of primal importance to determine the extent of queue backlogs to escape from network voids, and so, has a significant influence in the attained latency.*

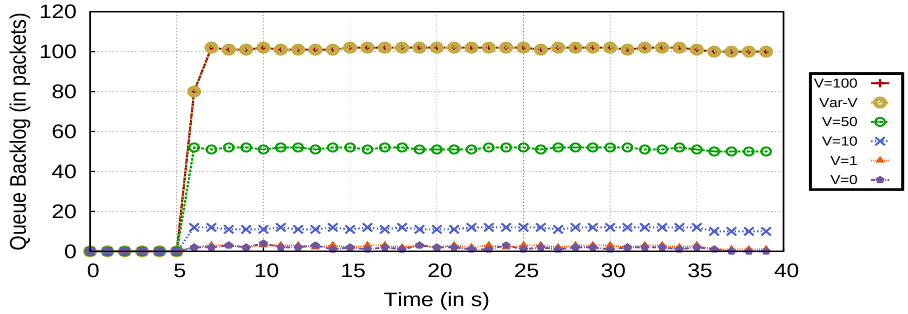

Figure 3: Impact of the value of the V parameter in the queue backlog to circumvent network voids. The void is circumvented once the queue backlog of the local minimum is stabilized, but some packets may get trapped.

Table 1: Impact of the value of the V parameter in latency.

Value of V parameter	Average Latency (ms)
V=0	143.81 ms
V=1	19.67 ms
V=10	126.79 ms
V=50	603.00 ms
V=100	1198.21 ms
Variable-V	1198.21 ms

Table 1 exhibits the consequent decrease on end-to-end latency with the decrease of the V parameter. Note that the case of the Variable-V, initialized to Q_{MAX}, requires to decrease its value up to half the value of Q_{MAX} to enable routing decisions based on the minimization of queue backlog differentials. The trend towards lower latencies has a turning point at V=0. Indeed, we observe a latency increase with respect to V=1 because the forwarding decisions are exclusively based on the minimization of queue backlogs, without taking into account the proximity to the destination. This results in long paths to reach the destination.

The third main aspect to highlight is that an excessive use of the minimization of queue backlogs to take routing decisions could result in excessive path lengths to escape from network voids. Figure 4 shows the hop distribution of all the packets carried in the backhaul with V=0. Note that the minimum path length in the considered case is four hops. Instead of merely using four hop paths, data packets traverse a number of hops that increases up to more than 60 hops. In particular, there is only a 20% of data packets following paths of a minimum number of hops. In this case, a fixed V value parameter set to 1 may solve the aforementioned problems for one single traffic flow. Nevertheless, as argued in [8] and [9], this is not a feasible solution, since it does not appropriately handle the traffic (presence of simultaneous flows) and network dynamicity that are the norm in wireless networks, hence resulting in routing loops and increased latencies.

Nonetheless, given its potential, the solution presented in this paper still relies on the Lyapunov drift-plus-penalty approach with a single queue per node to handle any-to-any traffic communication patterns. In this way, we benefit from its advantages, namely scalability, self-organization, statelessness, decentralization, and low control overhead.

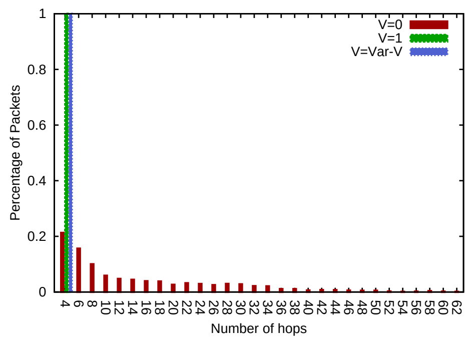

Figure 4: Histogram of the hop distribution when exclusively making forwarding decisions based on queue backlogs ($V = 0$).

4. Backpressure for Dynamic Small Cell Deployments

To address the above problems, this section describes the resulting scheme to manage both regular and sparse deployments. We propose to introduce a new mechanism in the routing penalty function that can handle network voids without incurring into additional routing recovery procedures, whose objectives are 1) to achieve reduced queue backlogs (and associated latencies), 2) to avoid excessive path lengths causing potential routing loops, and 3) to avoid packets to get trapped at data queues while maintaining the advantages of our original scheme [9].

Both the minimization of the Lyapunov drift and the Variable-V algorithm are key components to make the most out of the network resources. Furthermore, an appropriate cost function is fundamental to avoid the inefficiencies observed in section 3.2 for sparse deployments and achieve the aforementioned objectives. As revealed by the results in section 3.2, this is particularly noticeable when a substantial decrease of latency is observed when reducing the importance of the geographic-based cost function when taking routing decisions (low values of V). Subsections 4.1, 4.2, and 4.3 explain the main intuition, as well as the details of the operation and implementation of the cost function.

4.1. The Intuition behind the Cost Function

The cost function $c^d_{i,j}(t)$ conceived for sparse (and uniform) deployments follows the same trend of rewarding the selection of SCs closer to (and penalizing SCs farther from) the destination when there is uniform connectivity. However, the proposed cost function differs from the previous one designed for uniform SC deployments in two key points in order to avoid high queuing latencies in the presence of dead ends.

First, the cost function includes the possibility of rewarding routing decisions that select SCs located farther from the destination in the presence of dead-ends, rather than allowing packets to get trapped in data queues. Second, the cost function penalizes decisions generating 1-hop loops, which occur when a packet is routed back to the node from which the packet was just received. In this way, a 1-hop loop would occur when there is only one neighbor available and the Lyapunov drift minimization gains in importance regarding the cost function.

The results presented in section 5 show that the considered sparse deployments can serve the offered load appropriately when adding these two features in the routing cost function together with the geographic and backpressure components.

4.2. The Cost Function

Before delving into the details of the cost function, let us first define some auxiliary functions. Let the loop function $L_{i,j,d}(t)$ be equal to 1 when the current node i forwarding packet p received this packet from node j (that is, there is a 1-hop loop), and 0 otherwise. Additionally, let $NC_{i,d}(t)$ denote the set of 1-hop neighbors of node i closer to the destination d and $dist(n1, n2)$ be the function that calculates the Euclidean distance between nodes $n1$ and $n2$. Finally, let $OC_{i,d}(t)$ be a binary function that, when forwarding a packet from i headed to d, is equal to 1 if there is a single neighbor closer to d, and 0 otherwise.

According to this notation, the new proposed cost function is defined as:

$$c^d_{i,j}(t) = \begin{cases} L_{i,j,d}(t) - 1 & \begin{array}{l} dist(j,d) < dist(i,d) \\ \text{or } (dist(j,d) > dist(i,d) \\ \text{and } |NC_{i,d}(t)| = 0) \end{array} \\ & \\ 1 - 2L_{i,k,d}(t)OC_{i,d}(t) & \begin{array}{l} dist(j,d) > dist(i,d) \text{ and} \\ dist(k,d) < dist(i,d) \text{ and} \\ |NC_{i,d}(t)| \geq 1 \end{array} \end{cases} \tag{4}$$

For easing the description of the cost function, we use the sparse network ilustrated in Figure 5. The first case in equation (4) represents how the cost function treats neighbors i) closer to the destination, or ii) farther from the destination when there are not neighbors closer to the destination. For nodes closer to the destination, the loop function determines whether the cost is -1 (rewarding the closer neighbor if the loop function is 0), or is equivalent to 0 (base the decision on the queue drift to reward such neighbor if the loop is 1). In this sense, forwarding decisions approaching packets to the destination d are rewarded, unless it supposes a 1-hop loop. When packets reach dead-ends such as SC_{11} in Figure 5 (i.e., $|NC_{i,d}(t)| = 0$) our approach circumvents the void by rewarding forwarding decisions towards a node j, which is farther from d than the local node i. However, to avoid never-ending 1-hop loops, we check if the packet arrived from that same node j. This is controlled by the loop function, as when it is equal to 1, it makes the cost function towards node j equal to 0, and so, other nodes farther from d different from j are preferred. In terms of weights, these farther nodes are preferred over closer node j because i obtains a more negative cost function when forwarding this packet, and so, a higher weight is obtained. Since the packet is eventually forwarded to the neighbor with the highest weight at the time node i takes the forwarding decision, these nodes are selected first. On the other hand, if there is no other node except j for circumventing the void (i.e., going backwards through the same path), $L_{i,k,d}(t)$ is equal to 1, hence resulting in node i taking the forwarding decision exclusively based on the queue drift (e.g., backward path from SC_{24} to SC_{23} in Figure 5).

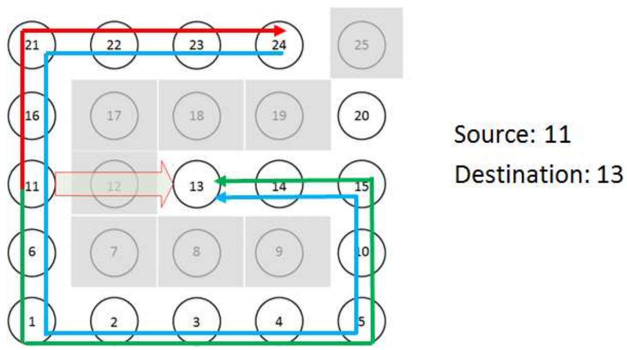

Figure 5: Topology surrounded by network void.

Since queue backlogs in dead-ends tend to be high, taking such decisions allows packets to go backwards to circumvent the void.

The second case of equation (4) is devoted to handle the cost calculation for nodes that are farther from the destination d when node i (the local node forwarding the packet) has neighbors closer to d. In the normal case, this will result in the cost being equal to 1. When combined with a positive value of V in equation 1, it results in lower weights than for those nodes close to d. Therefore, closer nodes are preferred. However, to avoid packets being trapped in the queues of dead-end nodes (or nodes close to dead-ends) another case must be handled.

For instance, the depicted network in Figure 5 forms a multi-hop line sub-topology surrounded by a network void. If a packet arrives to the dead-end node (e.g., SC_{11} in Figure 5), it will be handled by the first case, as explained above. However, once this packet reached the node just before the dead-end (e.g., SC_{16} or SC_6 in Figure 5), instead of sending it again towards the dead-end, it must send it backwards (see red arrow in Figure 5) until all the hops in the line sub-topology are traversed to be able to circumvent the whole. This is handled in the second case of the equation with the term $2L_{i,k,d}(t)OC_{i,d}(t)$. In fact, this term is different from 0 only when node i (the local node) receives a packet from node k (the only one closer to d) from which the packet arrived to i. In this case, both $L_{i,k,d}(t)$ and $OC_{i,d}(t)$ are equal to 1, which makes the cost become negative, which in turn, results in a high positive weight and the packet is sent to node j (farther from d) instead of k (closer to d). Thus, the packet traverses the line sub-topology in the backward direction. In this way, packet do not get trapped in the queues of such kind of sub-topologies.

4.3. Implementation details of the Cost Function

To implement the binary function $L_{i,j,d}(t)$, knowledge of an identifier of previous hop that forwarded the data packet is required. In the IP header of a data packet, there is neither information identifying the previous node/SC that transmitted a data packet nor the coordinates (or the IP address) of that node. Rather than adding new headers with the source IP address of the previous hop, in our implementation we use MAC addresses for that purpose.

In terms of state information, each SC maintains a table with information related to its available 1-hop neighbors. Furthermore, each entry of the table contains the queue backlog, the geographic coordinates, the IP address, and the MAC address of the neighbor.

Additionally, we store the MAC address of the node who forwarded each incoming packet stored in the data queue. For each data packet being forwarded, the SC checks whether the MAC address of the target next hop matches the MAC address stored with the packet. If there is a match, $L_{i,j,d}(t)$ becomes 1 for packet p, and in this way 1-hop routing loops are detected.

5. Evaluation

Section 5.1 describes the methodology followed, whereas section 5.2 and section discuss the results obtained.

5.1. Methodology

We conduct all the simulations with the ns-3 [11] network simulator, with a duration per simulation run of 50 seconds. The simulated network is a 5x5 square grid backhaul of SCs, where the distance between neighboring nodes is of 100 meters. The set of neighbors of a given SC are the nodes within a range of 100 meters. To carry backhaul traffic, every SCs has a single IEEE 802.11a WiFi interface configured to the same channel, and at a link rate of 54Mbps. In particular, we use a simple WiFi channel model with a 2-hop interference pattern that does not generate losses due to hidden nodes or propagation errors. The goal of all the experiments is to show the robustness of Backpressure routing for dynamic Small Cell deployments (BS) when some SCs of the modeled wireless mesh backhaul are unavailable. The set of unavailable SCs is selected as follows: an SC may be unavailable with a certain probability $p > 0$ when there is not any other SC already unavailable within wireless transmission range, else $p = 0$ (i.e., SC continues active). This methodology ensures that a path can be constructed between any possible combination of source-destination SCs. We fixed the set of powered off SCs to the 20% of the total number of SCs. Figure 2 illustrates an example of the strategy followed to switch off nodes in the 5x5 grid.

To evaluate BS robustness, simulation results compare the performance of BS with that of GPSR, and an idealized version of the shortest path routing algorithm (ISPA) under different traffic demands and different backhaul topologies. We use GPSR as a benchmark because it is the reference protocol in the literature [6] for comparison with geographic routing protocols, given its robustness and low control routing overhead. In turn, we use ISPA because it follows, in terms of data plane, the philosophy of current protocols deployed in the mobile backhaul such as MPLS-TP [5]. ISPA is 'ideal' in the sense of having a complete knowledge of the global network topology without exchanging any control information message. Therefore, ISPA always knows a priori the shortest path in terms of number of hops between any pair of nodes, hence building routes that do not use nodes that are switched off. That is, network voids are not a problem in ISPA. On the other hand, GPSR must change its operation from greedy forwarding mode to perimeter routing, using the right hand rule to guarantee that it will find a path to the destination. BS, on the contrary, merely uses the distributed computation of weights to circumvent network voids, as described in the previous section.

In addition to this, note that in all the ns-3 simulations, BS sends HELLO broadcast messages of 110 bytes every 100ms, whereas ISPA routing does not transmit any control messages. In turn, GPSR [12] sends HELLO messages of 135 bytes every 100ms, and includes an additional header in the data traffic, adding 50 bytes to the packet size (1488 bytes). Every SC maintains a single first-in first-out (FIFO) data queue of a maximum of 400 packets.

We characterized the performance of each protocol by measuring the throughput, latency, number of hops, and routing stretch (i.e., ratio of the hop count of the select paths to that of the shortest path) in every simulation in steady state (i.e., transient periods in each simulation run are discarded). These results are obtained using our implementation of BS, the GPSR implementation provided by the authors of [18], and the ISPA implementation found in [19]. Note that most of the results regarding throughput are omitted, since, unless explicitly mentioned in the paper, the offered load is fully served by the three routing protocols. For each of these network performance metrics, we generally used average values and boxplots to represent their statistical distribution. In particular, the box stretches from the 25th to the 75th percentiles, and the whiskers represent the 5th and 95th percentiles.

5.2. Impact of the Traffic Demand

This subsection provides the comparison of the performance of BS, GPSR, and ISPA while keeping a fixed set of SCs unavailable (see Figure 2) and considering different traffic workloads. In all the simulations, the same set of source-destination pairs are considered for all the routing protocols under comparison. The number of traffic flows injected to the network varies from 1 to 6, out of the set {1,2,4,6}. The generation of the set of traffic flows followed an incremental approach. That is, when moving from 2 to 4 flows, two new flows are added to the previous ones (i.e., 2 out of the 4 flows are the same ones as in the 2-flow case). Each of the four different traffic workload configurations were simulated forty times with different seeds. The use of different seeds provided the required random source/destination SC pairs. Each selected source SC injected 2Mbps

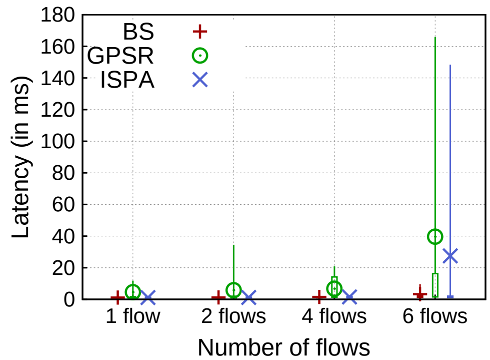

Figure 6: Average latency experienced by each injected workload of experiment in Section 5.2.

of UDP CBR traffic directed towards a destination SC, for a total offered load in the backhaul of $2Mbps \cdot Number\ of\ Flows$. Thus, we execute 160 different simulations for each protocol, hence resulting in 480 simulations in total. Figure 6 compares the average latency of BS, GPSR, and ISPA as the number of traffic flows varies from one to six flows.

With one, two and four traffic flows, the performance of BS is on average close to that of ISPA, as both protocols route data packets following paths of a minimum number of hops. Whilst ISPA builds offline end-to-end shortest path routing tables in every SC and requires topology information of the whole network to do that, BS only requires neighbor information. However, despite this remarkable qualitative difference, BS neither traverses paths with an excessive number of hops nor increases average queue backlogs. In turn, BS outperforms the latency values of GPSR given that the defined cost function in section 4 allows overcoming local minima in a more efficient way than GPSR. Actually, GPSR starts suffering from highly variable latencies when the number of flows is equal or bigger than two. The backhaul topology showed in Figure 2 and the randomly chosen source-destination SC pairs provoked GPSR to use perimeter mode for some combinations of source-destination pairs. When GPSR enters in recovery mode, the use of the right-hand rule to overcome a dead-end can often lead to suboptimal paths in terms of number of hops, increasing the end-to-end latency, even under light traffic conditions. Additionally, switching from greedy forwarding to recovery mode already increases the latency. Data packets experiencing recovery mode at the source node are queued and periodically served on bursts once the requested route, calculated using the right hand rule, is known [18]. In this way, when such packets are being served, the FIFO service policy is altered, which derives into additional latency perturbations to the rest of the traffic flows traversing the node.

When the number of traffic flows is equal to 6, the backhaul starts suffering from congestion and the latency increases with GPSR and ISPA because these protocols do not take into account all the available resources to take routing decisions. Thus, while certain SCs get congested, other SCs are not used at all. With six traffic flows, BS shows lower average latencies than GPSR and ISPA due to its inherent traffic distribution capabilities. By exploiting the minimization of the Lyapunov drift, BS attains an even resource consumption in the mesh backhaul. The resulting distribution of traffic is such that the resulting deviation from shortest path aims to relieve congestion, hence resulting in lower latencies despite traversing longer paths.

5.3. Impact of the Backhaul Topology

The goal of this subsection is to evaluate the robustness of BS compared to that of GPSR and ISPA when varying both the backhaul topology and the traffic demands. To this aim, this subsection extends the previous one to include in the evaluation twenty different mesh backhaul topologies. We generated each of the twenty topologies by varying the set of five SCs switched off from the 5x5 grid illustrated in Figure 2. The set of powered off SCs have been selected randomly following the strategy explained in subsection 5.1. Figure 7 plots the average latency distribution exhibited by BS, GPSR, and ISPA accounting the twenty sparse mesh setups as the workload increases (from one flow to six concurrent flows). In turn, the latency values have been calculated over forty independent repetitions. Each of these forty simulations runs used a random set of source-destination pairs. As a result, we run 3200 different simulations for each protocol, making a total of 9600 different simulations for the three routing variants.

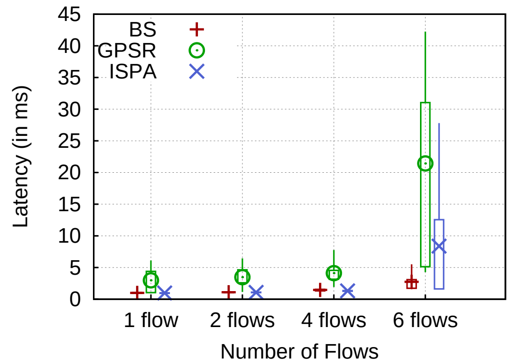

Figure 7: Average latency experienced for the different backhaul topologies and and injected workload of experiment in Section 5.3

The most remarkable observation is that simulation results confirm the robustness of BS in different sparse mesh backhaul topologies. BS outperforms both GPSR and ISPA showing a reduction in terms of average latency of up to a 85% and 70% respectively, while maintaining its inherent load balancing capabilities and overcoming voids with a low routing stretch. Despite varying twenty times the backhaul topology, the latency trend showed by the three protocols in Figure 7 is similar to that showed in the previous subsection with a fixed backhaul topology (see Figure 6). The workload is almost always served, yet there are some throughput inefficiencies under heavy traffic conditions, i.e. with six traffic flows.

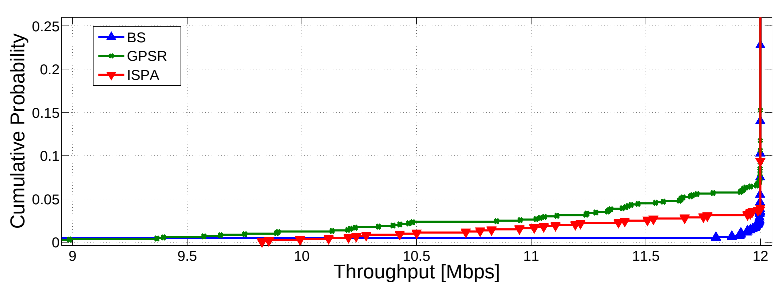

Figure 8: Aggregated throughput for six traffic flows.

Figure 8 shows the cumulative distributed function of the attained throughput for six traffic flows. Results show that GPSR and ISPA fail more frequently than BS, but for a minority (less than 1%) of the chosen source-destination pairs. In these cases, where the workload is above the rates that the network can handle, BS exhibits a remarkable degradation of throughput. This is due to the fact that the distributed variable-V algorithm of BS excessively decreases the V values in most of the SCs to zero. Thus, routing decisions are merely focused on minimizing the Lyapunov drift rather than maximizing the rate of data arrivals at the destinations. The other protocols stick to a given route, handling such saturation conditions by losing packets due to buffer overflows at nodes. This behavior results in less packets being transmitted over the network, and so, the remaining packets can be more appropriately served. However, the reader should recall that these are saturation conditions that the operator will avoid by other means. One potential solution is the design of a distributed flow rate controller that shapes the injected traffic to prevent the network from reaching saturation as in [20]. Nevertheless, this is out of the scope of this paper.

Figure 9 depicts the average path length distribution of the three routing variants for the twenty sparse mesh setups and a workload of four traffic flows. Note that ISPA represents the lower bound in terms of path length distribution, and can be considered the optimal routing solution in terms of maximizing throughput and minimizing latency under light traffic loads. Interestingly, Figure 9 confirms that BS exhibits a path length distribution close to that attained by ISPA for the twenty sparse mesh setups and different combinations of four traffic flows injected in the network.

Figure 10 illustrates the specific distribution of latency of the three routing variants for each of the twenty backhaul mesh topologies and a workload of four traffic flows. We can highlight two main observations. First, BS clearly outperforms GPSR for the twenty mesh backhaul topologies given the potential inefficiency of the routing recovery mode of GPSR. Second, BS presented similar latencies in most of the mesh backhaul topologies under evaluation compared to ISPA. We only found one backhaul topology, labeled as T20, of noticeable higher mean average latency in BS compared to ISPA. For these specific setup, BS experiences a higher latency in the 5% of the forty repetitions.

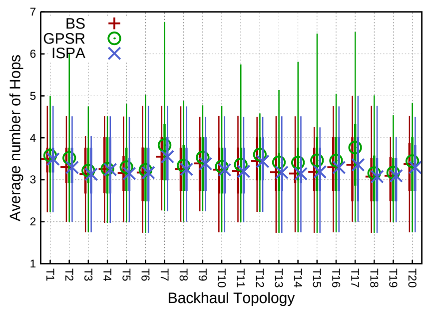

Figure 9: Average number of hops experienced in each one of the 20 mesh backhaul topologies with a workload of 4 flows

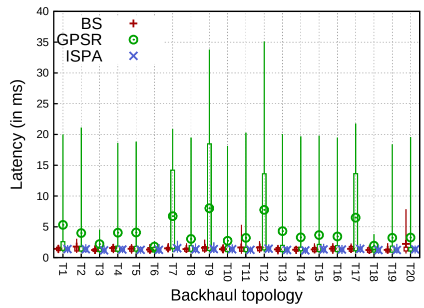

Figure 10: Latency distribution in each one of the 20 mesh backhaul topologies with a workload of 4 flows

Figure 11 presents the attained latency distribution attained by BS, GPSR, and ISPA for the considered twenty backhaul topologies and a workload of six traffic flows. BS clearly outperforms GPSR both on the average values and on their variance. Additionally, latency values of BS present a better trend on average than those attained by ISPA, as showed in Figure 7. The single shortest path selected by ISPA is not sufficient to serve efficiently the traffic, causing congestion. Load balancing is required to avoid packets stay longer periods at SC queues. Indeed, the problem with GPSR as well as ISPA is that they are insensitive to congestion, being not able to adapt dynamically their routes to relieve traffic congestion. The improvement in terms of latency experimented by BS is explained in part by its ability to distribute traffic, exploiting in a more efficient way the backhaul resources.

Despite the increase of the offered load to six traffic flows, BS still shows a path length distribution similar to that of ISPA. This indicates that BS leverages multiple non congested paths of a low number of hops, whereas ISPA merely uses one single shortest path. This explains why the difference in number of hops is so small between ISPA and BS. It is also important to note that the path length distribution of BS proves that routing loops are rare for most of the

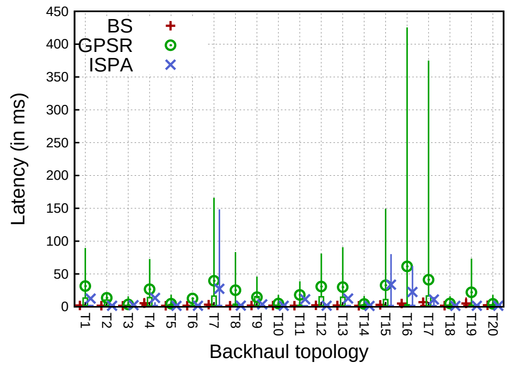

Figure 11: Latency distribution in each one of the 20 mesh backhaul topologies with a workload of 6 flows

twenty topologies under consideration. When balancing traffic, the aim of BS is to prioritize short rather than long paths. Figure 12 presents the CDF of the routing stretch metric for the BS and GPSR protocols with respect to ISPA (the routing protocol providing the optimal route in terms of hops) when injecting a workload of 6 flows. As we can observe, GPSR experiences path lengths up to 4.5 times bigger than ISPA, whereas BS path lengths do not exceed 1.7 times this value. BS incurs in more hops in order to exploit the resources of the network when overcoming a communication void, while GPSR uses the right-hand rule, which is not always effective. Simulation results indicate that the right-hand rule chooses on average a non optimal route in 50% of the cases, which derives in a high routing stretch value.

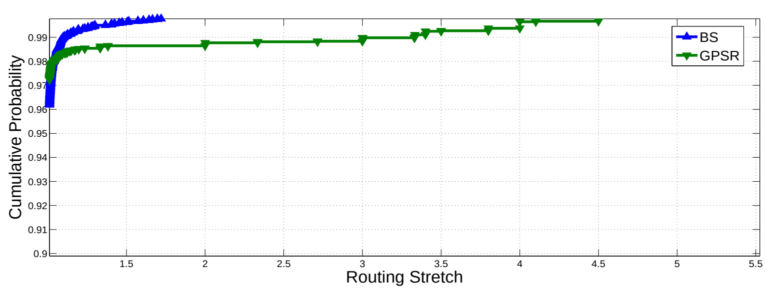

Figure 12: Routing stretch CDF of the flows considered in each one of the 20 mesh backhaul topologies with a workload of 6 flows

6. Conclusions

This paper presents BS, a self-organized routing protocol that combining backpressure, geolocation, and information carried in the packet avoids excessive network latencies while serving offered traffic demands under dynamic SC deployments. Amongst others, BS features self-organization, scalability, decentralization, (quasi-) statelessness, and low control overhead.

Simulation results with ns-3 show that BS provides robustness across a wide variety of wireless backhaul topologies and traffic demands, obtaining a similar (or even better) performance compared to ideal shortest path routing algorithm (ISPA), a protocol abstracting the properties of legacy transport network level protocols (e.g., MPLS-TP). BS experiences similar latencies to ISPA for light loads, and exhibits a latency reduction on average of up to a 70% under more heavy traffic demands due to its inherent load balancing capabilities. Besides, BS outperforms GPSR, the reference protocol for geographic routing, showing a reduction of up to 85% in average latency, while serving all the offered load. The key reasons for such latency improvement are the use of multiple paths when network congestion requires it, without incurring into a high value of the routing stretch metric (i.e., 1.7 times the shortest path). Overall, we believe that the results presented in this paper are crucial to consider maintaining the benefits of backpressure routing showed in uniform deployments for dynamic small cells deployments.

7. Acknowledgments

This work was supported by the TEC2011-29700-C02-01 grant of Spanish Ministry of Economy and Competitiveness and by the 2009-SGR-940 grant of the Catalan Government.

References

[1] W. Webb, Wireless Communications: The Future, Wiley, 2007.

[2] J. Ferragut, J. Mangues-Bafalluy, J. Núñez Martínez, F. Zdarsky, Traffic and mobility management in networks of femtocells, ACM/Springer Mobile Networks and Applications Journal 17 (5) (2012) 662–673.

[3] U. Paul, A. P. Subramanian, M. M. Buddhikot, S. R. Das, Understanding traffic dynamics in cellular data networks, in: Proc. of the 30th INFOCOM, IEEE, 2011, pp. 882–890.

[4] Meeting the 1000x challenge: The need for spectrum, technoogy, and policy innovation, `http://www.4gamericas.org`.

[5] M. Bocci, A framework for mpls in transport networks, Internet RFC 5921.

[6] F. Cadger, K. Curran, J. Santos, S. Moffett, A survey of geographical routing in wireless ad-hoc networks, IEEE Communications Surveys Tutorials 15 (2013) 621–653. `doi:10.1109/SURV.2012.062612.00109`.

[7] D. Chen, P. Varshney, A survey of void handling techniques for geographic routing in wireless networks, IEEE Communications Surveys Tutorials 9 (1) (2007) 50–67. `doi:10.1109/COMST.2007.358971`.

[8] J. Núñez Martínez, J. Mangues-Bafalluy, M. Portoles-Comeras, Studying practical any-to-any backpressure routing in wi-fi mesh networks from a lyapunov optimization perspective, in: Proc. of 8th IEEE Int. Conf. on Mobile Adhoc and Sensor Systems (MASS), 2011.

[9] J. Núñez Martínez, J. Mangues-Bafalluy, Distributed lyapunov drift-plus-penalty routing for wifi mesh networks with adaptive penalty weight, in: Proc. of the 13th IEEE International Symposium on World of Wireless, Mobile and Multimedia Networks (WoWMoM), 2012, pp. 1–6. `doi:10.1109/WoWMoM.2012.6263779`.

[10] J. Núñez-Martínez, J. Baranda, J. Mangues-Bafalluy, Backpressure routing for the backhaul in sparse small cell deployments, in: Proc. of the 32nd IPCCC, IEEE, 2013, pp. 1–2.

[11] The ns-3 network simulator, available at: http://www.nsam.org.

[12] B. Karp, H.-T. Kung, GPSR: Greedy perimeter stateless routing for wireless networks, in: Proc. of the 6th annual international conference on Mobile computing and networking, ACM, 2000, pp. 243–254.

[13] L. Tassiulas, A. Ephremides, Stability properties of constrained queueing systems and scheduling policies for maximum throughput in multihop radio networks, IEEE Trans. on Automatic Control 37 (12) (1992) 1936–1948.

[14] L. Bui, R. Srikant, A. Stolyar, A novel architecture for reduction of delay and queueing structure complexity in the back-pressure algorithm, IEEE/ACM Transactions on Networking 19 (6) (2011) 1597–1609. doi:10.1109/TNET.2011.2126593.

[15] L. Ying, S. Shakkottai, A. Reddy, S. Liu, On combining shortest-path and back-pressure routing over multihop wireless networks, IEEE/ACM Transactions on Networking 19 (2011) 841–854.

[16] M. J. Neely, Stochastic Network Optimization with Application to Communication and Queueing Systems, Synthesis Lectures on Communication Networks, in Morgan & Claypool Publishers, 2010.

[17] S. Moeller, A. Sridharan, B. Krishnamachari, O. Gnawali, Routing without routes: the backpressure collection protocol, in: Proc. of the 9th ACM/IEEE Int. Conf. on Information Processing in Sensor Networks (IPSN), ACM, New York, NY, USA, 2010, pp. 279–290. doi:10.1145/1791212.1791246.

[18] A. Fonseca, A. Camoes, T. Vazao, Geographical routing implementation in ns3, in: Proc. of the 5th Int. ICST Conf. on Simulation Tools and Techniques, ACM, 2012, pp. 353–358.

[19] J. Kim, Shortest Path Routing Protocol for NS3[Online; accessed 11-July-2014].

[20] R. Laufer, T. Salonidis, H. Lundgren, P. Le Guyadec, Xpress: a cross-layer backpressure architecture for wireless multi-hop networks, in: Proc. of the 17th Annual Int. Conf. on Mobile Computing and Networking (MobiCom), ACM, 2011, pp. 49–60.