This article has been accepted for publication in IEEE/ACM Transactions on Networking. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TNET.2023.3273349

Distributed Resource Allocation and Flow Control algorithms for mmWave IAB Networks.

Swaroop Gopalam, Stephen V. Hanly and Philip Whiting
School of Engineering, Macquarie University, Sydney, Australia
{swaroop.gopalam, stephen.hanly, philip.whiting}@mq.edu.au

***Abstract*—This paper presents a new distributed slot reservation frame-work for joint resource allocation and flow control in mmWave IAB networks. We derive the Dynamic Slot Reservation (DSR) algorithm from a novel approach to solve a minimum clearing time linear program in a completely distributed manner. The algorithm to solve this problem, the Static Slot Reservation (SSR) algorithm, is also a contribution of the paper. We compare the delay performance of the DSR algorithm with a well known optimal, centralized algorithm, the joint-MWM algorithm [1], [2], for a realistic IAB network scenario of multi-hop flows. We show that flows that traverse several links have significantly lower delays under DSR than under the joint-MWM algorithm. This paper also provides an instantaneous rate control policy for IAB networks which changes flow rates based on the number of flows at each node in the network. The flow rates under this policy are the same as the steady-state flow rates achieved by the DSR algorithm. We prove that the proposed flow control policy provides stability for all flow arrival rate vectors that are achievable by any flow control policy. This paper provides distributed admission control policies to provide rate and/or latency guarantees to flows under dynamic scenarios with stochastic flow arrivals and changing access link rates.**

I. Introduction

mmWave cellular networks are expected to play a key role in the next generation wireless communications (5G) [3]. They are capable of delivering very high rates, due to the vast amount of spectrum available in the mmWave band. However, wireless communication at mmWave frequencies comes with two major obstacles, including high isotropic propagation loss and sensitivity to blockage by the objects in the environment. As a result, ultra dense deployments of Next Generation Node Bases (gNBs) with small cell sizes and directional communication using beam-forming, are being considered to provide universal coverage.

Under such deployments, it is prohibitively expensive to provide fibre backhaul support to all the gNBS. Hence, there is a growing interest in multi-hop relaying (or self backhauling) for mmWave cellular networks. Notably, as part of its standardization efforts, 3GPP has completed a recent study item on the potential solutions for efficient operation of integrated access and wireless backhaul (IAB) for NR [4]. The study emphasizes the joint consideration of mmWave radio-access and backhaul for mmWave cellular networks. The ability of gNBs to point very narrow, high gains beams, and the high bandwidth, makes mm-wave an attractive band in which to integrate access and backhaul.

In this paper, we consider a multi-hop IAB network, where a fraction of gNBs are deployed with dedicated fiber backhaul links, referred to as IAB donors [4]. The other gNBs (referred to as IAB nodes) relay their backhaul data over wireless mmWave links, possibly in multiple hops to an IAB donor. According to [4], an IAB node establishes a link to a parent node (either another IAB node or a donor) and the central unit (CU) at the IAB donor establishes a forwarding route to the IAB node via the parent. Therefore, the traffic of a UE is forwarded along this established route from the IAB donor to the UE (in downlink). We focus on the spanning tree (ST) topology from [4], where each IAB node has exactly one parent node (either a IAB node or the IAB donor). An example of such an IAB network can be seen in Figure. 1(a).

We address two major challenges of multi-hop IAB networks in this paper, 1) Distributed flow control and resource allocation among wireless and backhaul links in an in-band backhauling scenario (*i.e.* access and backhaul links use the same frequency band) and 2) Providing end-to-end Quality of Service (QoS) guarantees for UEs. In an in-band scenario, as we consider in this paper, there are constraints on simultaneous link activation due to half-duplex communication. Hence, link activation (and time slot allocation) has to be coordinated across the network to avoid conflicts. Secondly, the traffic of a UE may have to be forwarded along several backhaul links due to the multi-hop nature of the IAB network. Hence, the end-to-end flow rate of a UE depends on the congestion level at each link in the path from IAB donor to the UE. Ensuring end-to-end QoS for existing UEs is a challenging problem as new UEs arrive into the network.

The first challenge, 1) above, is part of the general challenge of joint flow control and resource allocation in wireless multi-hop networks [1], [2]. In wireless networks, this challenge is inherently *cross-layer*, since resource allocation must be done in conjunction with finding the optimal flow rates. The classical Network Utility Maximization (NUM) approach to optimal cross-layer control utilizes the max-weight algorithm [5] for resource allocation [1], [2]; this involves finding a maximum weight feasible set of links, which is NP hard in general. Recent work has shown that the particular tree structure of IAB networks makes this max-weight problem quite simple [6], but still it requires messages to traverse the network each time a scheduling decision is made, *i.e.* in each slot.

This research was supported in part by the Australian Research Council under Discovery Project DP180103550. It was also supported by a iMQRTP PhD scholarship from Macquarie University.

In this paper, we take a different approach that results in a cross-layer algorithm that is completely distributed and which does not require finding a network-wide, maximum weight feasible set of links in each slot. We propose a distributed joint flow control and resource allocation algorithm for IAB networks, the Dynamic Slot Reservation (DSR) algorithm. The DSR algorithm is of low complexity and can be implemented locally at each gNB in the network. We show that it only requires communication with the neighbors of a gNB. The algorithm allows for easy extension of the IAB network by addition of new gNBs, since it is implemented locally at each gNB. We also show that admission control can be incorporated in the DSR algorithm to provide QoS guarantees for end-to-end rate and latency in a distributed manner.

Our new approach is based around solving a minimum clearing time linear program (3) (formulated in Section III) for slot/resource allocation. We show how to achieve any feasible rate vector (within the achievable rate region) using the solution to the linear program. We develop the *Static Slot Reservation* (SSR) algorithm and show that it provides a completely distributed approach to solving the minimum clearing time linear program. The SSR algorithm can be described as a *book ahead* slot reservation system, where a link l (updating at slot t) is allowed to reserve a number, τ_l, of future slots, which are slots that occur at times later than t. The choice of slots reserved by l is such that they do not overlap with the existing reserved slots of conflicting links.

The SSR algorithm performs resource allocation by slot reservation for a static setup of UEs with fixed rates. We develop our approach further in the *Dynamic Slot Reservation* (DSR) algorithm, by jointly controlling rates and slot reservation in a dynamic IAB network with stochastic UE arrivals and departures. The control of flow rates is achieved by adjusting the τ_l (numbers of reserved slots) values. The τ_l values are decided based on the local information at link l. We compare the delay performance of the DSR algorithm with a well known optimal, centralized cross-layer algorithm, the joint-MWM algorithm [1], [2], for a realistic IAB network scenario of multi-hop flows. We show that flows that traverse several links have significantly lower end-to-end delays under DSR than under the joint-MWM algorithm.

Our contributions are as follows:

- We propose a new distributed slot reservation framework, the Static Slot Reservation Algorithm, for solving a minimum clearing time linear program for a deterministic network setup. This is the book ahead slot reservation scheme described in Algorithm 1.
- We show that the window size $w(t)$ (defined in (6)) under the SSR algorithm is non-increasing in Theorem 2 and that it converges to an optimal solution, *i.e.* the least possible value for window size, within a fixed number of slots which is linear in the number of links, in Theorem 3. These results establish that the SSR algorithm achieves all vectors inside the achievable rate region.
- In Section VI, we introduce a dynamic stochastic model of UE arrivals and departures, and extend the Static Slot Reservation Algorithm to the dynamic scenario, which becomes Algorithm 3, the DSR Algorithm. This algorithm is distributed, using only local information.
- In Section VII, we show that the DSR Algorithm has comparable performance to a classical centralized algorithm, the joint-MWM algorithm from [1], [2], with better performance for flows with a larger number of hops.
- In Section VIII, we consider a stationary flow control policy which instantaneously achieves the steady state flow rates of the DSR Algorithm, changing flow rates every time that the network state changes. In Theorems 5 and 6, we characterize the arrival rate vectors for which ergodicity of the state Markov process is possible under some stationary flow control policy. Theorem 6 shows that the network state Markov process is ergodic under the proposed flow-control policy for all the arrival rate vectors where ergodicity is possible.
- In Section IX, we derive the flow rates received by each UE as a function of the network state in Theorem 7. We use this result to propose a distributed admission control scheme which provides guaranteed QoS.

A. Related Work

The NUM approach for internet congestion control was introduced in [7]. The framework was applied for congestion/flow control in the internet and was used to derive network capacity in [8]–[10]. The NUM framework was extended to multi-hop wireless networks in [1], [2], [11], where cross-layer algorithms for joint flow control, routing and scheduling were developed. In the wireless multi-hop networks, the scheduling component requires the centralized max-weight algorithm for optimality [5].

System level analysis of mmWave IAB networks with full-duplex and half-duplex self backhualing was recently considered in [12], under a NUM framework with latency and flow rate constraints. The rate and latency gains for full-duplex over half-duplex were derived for a similar model of IAB tree network, but for a static setup of flows. For mmWave IAB neworks, the following works have taken an optimization approach for flow control and resource allocation [13]–[19], [19]–[22], where only centralized solutions and a static setup of flows were considered. For a full duplex IAB network, a weighted proportional fair scheduler, along with an ad-hoc flow control algorithm was proposed in [13].

For general mmWave multi-hop networks (*i.e.* not based on the 3GPP IAB architecture), the NUM framework for joint congestion control and scheduling was considered in [23], [24], which also relies on the max-weight algorithm. For general mmWave multi-hop networks, optimization of joint routing and resource allocation was considered in [14]–[20], and utility maximization was used for path selection and scheduling in [19], [21], [22]. These works have also only considered centralized solutions and a static setup.

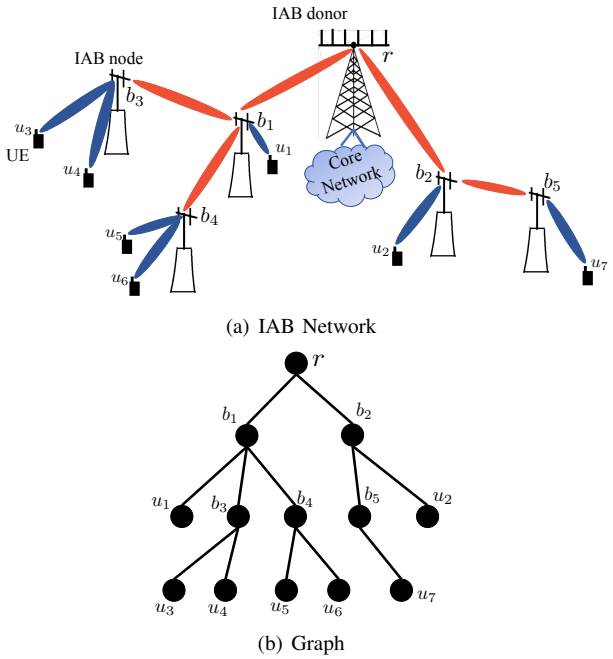
B. Organization of the paper

In Section II, we provide the system model for a static IAB network with a fixed set of UEs and with fixed data rates. In Section III, we propose a notion of rate vector achieveability

This article has been accepted for publication in IEEE/ACM Transactions on Networking. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TNET.2023.3273349

and introduce a minimum clearing time resource allocation problem for this deterministic setup. In Section IV, we present the Static Slot Reservation Algorithm, and in Section V, we show that it converges to the solution of the minimum clearing time linear program. In Section VI, we introduce the dynamic system model with stochastic UE request arrivals and departures, and present the Dynamic Slot Reservation (DSR) Algorithm. In Section VII, we present numerical results. In Section VIII, we provide stability results for a policy which instantaneously realizes the steady state flow rates of the DSR algorithm. In Section IX, we provide distributed admission control policies which can be implemented in conjunction with the DSR algorithm to provide end-to-end QoS guarantees. In Section X, we provide conclusions and discuss future work.

II. System Model



(a) IAB Network

(b) Graph

Fig. 1. IAB Network and its graph representation

Consider a downlink multi-hop IAB network where $\mathcal{B}$ is the set of gNBs and $\mathcal{U}$ is the set of UEs, *e.g.* see Fig. 1(a). Here, $r \in \mathcal{B}$ is the IAB donor which has a fibre backhaul connection, and the other gNBs relay their backhaul data from r to the UE, over the mmWave links. We consider the tree topology where the backhaul data of each gNB $b \in \mathcal{B} - \{r\}$ is routed from a unique parent node $p(b) \in \mathcal{B} - \{b\}$. We represent the network using a rooted tree $G = (\mathcal{V}, \mathcal{L}, r)$, where $\mathcal{V} = \mathcal{U} \bigcup \mathcal{B}$ is the set of nodes, $\mathcal{L}$ is the set of wireless links, including gNB-gNB backhaul links and gNB-UE access links. For example, Fig. 1(b) shows the graph representation for the IAB network in Fig. 1(a).

We consider a slotted model where $t \in \mathbb{Z}_+$ represents the t th slot. For a link $\ell \in \mathcal{L}$, we define the *allocation* of ℓ in slot t, $U_\ell(t)$, as follows. $U_\ell(t) = 1$, if slot t is allocated to ℓ. Otherwise, $U_\ell(t) = 0$. There are constraints on link activation as follows. Let $\mathcal{L}_b$ denote the set of downstream links of a gNB $b \in \mathcal{B}$. 1) Each gNB b is allowed to transmit on at most one of its downstream links in $\mathcal{L}_b$ in a given time-slot, *e.g.* In Fig. 1(b), r is only allowed to transmit on either link (r, b_1) or link (r, b_2) in a given time-slot. 2) Half-duplex constraint: the backhaul link ℓ_b connecting b and $p(b)$, cannot be activated at the same time as a link in $\mathcal{L}_b$. Under these contraints, links l_1 and l_2 cannot be active at the same time if they share a common node (this is known as the node-exclusive interference model [1], [2]). For each $\ell \in \mathcal{L}$, let $I(\ell)$ denote the set of links which cannot be active at the same time as ℓ. The scheduling constraints are given by the following equation.

$$U_\ell(t) U_m(t) = 0, \ \forall m \in I(\ell), \ t \in \mathbb{N} \tag{1}$$

We initially consider a snap-shot model, where each link $\ell \in \mathcal{L}$ is associated with a fixed instantaneous rate R_ℓ (in bits per slot). Each UE $u \in \mathcal{U}$ is associated with a fixed traffic rate α_u (in bits per slot), which is determined by a flow control policy at the root gNB r.

III. Minimum Clearing Time Problem Formulation

In this section, the definition of a resource allocation algorithm, and what it means for a traffic rate vector to be achievable under the algorithm are given. We formulate the minimum resource clearing problem as a linear program (LP). The solution of the LP provides a characterization of the achievable region, *i.e.* set of all traffic rate vectors $\boldsymbol{\alpha} := [\alpha_u]_{u \in \mathcal{U}}$ which are achievable under some allocation algorithm.

Definition 1. *A resource allocation algorithm is characterized by a set of $|\mathcal{L}|$ allocation sequences, $U_\ell \in \{0,1\}^\infty$ for each $\ell \in \mathcal{L}$, such that $U_\ell(t) U_m(t) = 0, \forall \ \ell \in \mathcal{L}, m \in I(\ell)$, for each $t \in \mathbb{N}$.*

Let P_u denote the set of links in the path from $u \in \mathcal{U}$ to root r in G. For $\ell \in \mathcal{L}$, we define $\gamma_\ell := \sum_{u: \ell \in P_u} \alpha_u / R_\ell$ as the target utilization of link ℓ, which is the fraction of time ℓ must be active to satisfy the traffic rate vector $\boldsymbol{\alpha} := [\alpha_u]_{u \in \mathcal{U}}$. We assume that $\boldsymbol{\gamma} := [\gamma_\ell]_{\ell \in \mathcal{L}} \in \mathbb{Q}^{|\mathcal{L}|}$, *i.e.* γ_ℓ is a rational number for each $\ell \in \mathcal{L}$.

Definition 2. *The rate vector $\boldsymbol{\alpha}$ is achievable under a resource allocation algorithm, with allocation sequences $U_\ell \in \{0,1\}^\infty$, for each $\ell \in \mathcal{L}$, if and only if*

$$\liminf_{T \to \infty} \frac{1}{T} \sum_{t=0}^{T} U_\ell(t) \geq \gamma_\ell \tag{2}$$

for each $\ell \in \mathcal{L}$.

A. Achievable Rate Region

We now characterize the achievable rate region, the region of traffic rate vectors that are achieved by resource allocation algorithms. We begin by formulating the Minimum Clearing Time Linear Program.

Definition 3. *A feasible set $S \subseteq \mathcal{L}$ is a set of links such that for each $\ell \in S$, $I(\ell) \bigcap S = \phi$.*

Note that the scheduled set of links $\{\ell \in \mathcal{L} : U_\ell(t) = 1\}$ in each slot t is feasible, under a resource allocation algorithm.

We consider the most efficient allocation of slots, which requires the minimum slots/resources, to achieve $\boldsymbol{\gamma}$ (and the corresponding traffic rate vector $\boldsymbol{\alpha}$) as the minimum clearing time problem in the following LP (3).

$$\begin{aligned} \min \sum_{S\in\mathcal{S}} f_S \\ \text{s.t.} \sum_{S:\ell\in S} f_S \geq \gamma_\ell, \ell \in \mathcal{L}; \\ f_S \geq 0, \forall S \in \mathcal{S} \end{aligned} \tag{3}$$

where $\mathcal{S}$ is the set all feasible sets, and f_S is the long-term fraction of slots allocated to a feasible set S.

We can construct a resource allocation policy, using an optimal solution $[f_S^*]_{S\in\mathcal{S}}$, as follows. Since $\boldsymbol{\gamma} \in \mathbb{Q}^{|\mathcal{L}|}$, it follows that there exists a rational optimal solution $[f_S^*]_{S\in\mathcal{S}} \in \mathbb{Q}^{|\mathcal{S}|}$ of LP (3). Let $\mathcal{S}'$ denote the set of $S \in \mathcal{S}$ such that $f_S^* > 0$. It is clear that a strictly positive scaling factor $\tau \in \mathbb{N}$ exists, such that τf_S^* is a natural number for each $S \in \mathcal{S}'$.

For the sake of convenience, we let $[t_1 : t_2]$ represent the set $\{t_1, \ldots, t_2\}$ for $t_2 \geq t_1$. Consider the following policy which uses blocks of length $B := \tau \sum_{S\in\mathcal{S}} f_S^*$ slots to allocate to the sets $S \in \mathcal{S}'$. Let $\{S_i\}_{i=1}^{|\mathcal{S}'|}$ be an ordering of set $\mathcal{S}'$. The first $\tau^* f_{S_1}$ slots, $[(i-1)B+1 : (i-1)B+\tau f_{S_1}^*]$, in the i-th block are allocated to links in set S_1, for $i \in \mathbb{N}$. The next slots in the block are allocated in order for $j = 2, \ldots, |\mathcal{S}'|$; *i.e.* slots

$$\left[(i-1)B + \tau \sum_{k=1}^{j-1} f_{S_k}^* \ : \ (i-1)B + \tau \sum_{k=1}^{j} f_{S_k}^*\right]$$

in the i-th block are allocated to each link in set S_j.

From construction, it is clear that each link $\ell \in \mathcal{L}$ receives a long-term fraction $\sum_{S:\ell\in S} f_S^* / \sum_{S\in\mathcal{S}} f_S^*$ of slots, *i.e.*

$$\lim_{T\to\infty} \sum_{t=1}^{T} U_\ell(t)/T = \frac{\sum_{S:\ell\in S} f_S^*}{\sum_{S\in\mathcal{S}} f_S^*} \tag{4}$$

If the optimal solution, $(f_S^*)_{S\in\mathcal{S}}$, of the LP (3) satisfies $\sum_{S\in\mathcal{S}} f_S^* \leq 1$, then $\sum_{S:\ell\in S} f_S^* / \sum_{S\in\mathcal{S}} f_S^* \geq \gamma_\ell$ for each $\ell \in \mathcal{L}$. Hence from (4), the traffic rate vector $\boldsymbol{\alpha}$ is achieved by the proposed policy. In Theorem 1, we show that $\sum_{S\in\mathcal{S}} f_S^* \leq 1$ is also a necessary condition for achievability, thus characterizing the achievable region.

Theorem 1. *The traffic rate vector $\boldsymbol{\alpha}$ is achievable, if and only if the optimal value of LP (3), $f^* := \sum_{S\in\mathcal{S}} f_S^* \leq 1$.*

Proof. See Appendix A. □

Solving LP (3) directly involves gathering the necessary information at a central node. Moreover, the number of feasible sets $|\mathcal{S}|$ under consideration, grows exponentially in the size of the network. In this paper, we show that problem (3) is not intrinsically hard. In fact, we provide a distributed resource allocation algorithm which requires at link ℓ only local information from the neighboring links in $I(\ell)$. We show that the algorithm converges to an optimal solution of LP (3), after a number of slots that is linear in the size of the network.

IV. Static Slot Reservation Algorithm

For the resource allocation algorithm, we consider a load $\tau_\ell := \gamma_\ell \tau_s$ on each link ℓ, by fixing a scaling factor $\tau_s \in \mathbb{N}$[1]. The scaling factor τ_s is chosen such that τ_ℓ is a positive integer for each $\ell \in \mathcal{L}$, which is possible since γ_ℓ is a rational number for each ℓ. We treat τ_ℓ as the number of slots required by ℓ.

The algorithm can be described as a *book ahead* slot reservation scheme, where a link $\ell \in \mathcal{L}$, updating at slot t, is allowed to reserve τ_ℓ future slots (*i.e.* slots $> t$). The choice of slots has to be made such that 1) they form a contiguous block, *i.e.* $[t' : t' + \tau_\ell - 1]$ for some $t' > t$; 2) they do not overlap with any of slots currently reserved by the conflicting links in $I(\ell)$. We refer to the first slot t' in the block as a *starting point*, and the last slot $t' + \tau_\ell - 1$ as an *ending slot*.

It is possible to describe the allocations using only starting points, due to contiguous slot bookings. Hence, the proposed algorithm satisfies the following Property 1.

Property 1. *For each $\ell \in \mathcal{L}$, there exists a sequence of starting points $\{s_\ell^n\}_{n=1}^\infty \subseteq \mathbb{N}$ which determine the allocation sequence U_ℓ as follows.*

$$U_\ell(t) := \begin{cases} 1 & \text{if } t \in \bigcup_{n=1}^\infty [s_\ell^n : s_\ell^n + \tau_\ell - 1] \\ 0 & \text{otherwise.} \end{cases} \tag{5}$$

We present our distributed slot reservation algorithm, the *Static Slot Reservation* (SSR) algorithm in Algorithm 1. In the algorithm, we use the notation $T_\ell(t)$ to denote the set of currently booked slots for link ℓ at time t. In lines 1-7 of Algorithm 1, a set of feasible starting points s_ℓ^1 are calculated for the initial bookings of slots. Links are ordered arbitrarily, and the initial blocks of slots are allocated in this order. The allocation to each link, in line 6, takes into account the interference constraints from the links which have already been allocated (*i.e.* earlier in the ordering). Given the initial starting points $\{s_\ell^1\}_{\ell\in\mathcal{L}}$, the update rule (in lines 10-22) generates the subsequent starting points $\{s_\ell^n\}_{n=2}^\infty$ for each $\ell \in \mathcal{L}$, as discussed further below.

In Algorithm 1, the next slot booking for a link is updated at the end of the current booking; *e.g.* for link ℓ, with current booking ending at slot $t_1 = s_\ell^n + \tau_\ell - 1$, the next booking is $T_\ell(t_1 + 1) = [s_\ell^{n+1} : s_\ell^{n+1} + \tau_\ell - 1]$. In between the booked time blocks, for $t \in [s_\ell^n + \tau_\ell : s_\ell^{n+1} - 1]$, the currently booked slots do not change: $T_\ell(t) = T_\ell(t_1 + 1)$. During the booked time block, for $t \in [s_\ell^{n+1} : s_\ell^{n+1} + \tau_\ell - 1]$, the currently booked slots are decremented according to: $T_\ell(t) = [t : s_\ell^{n+1} + \tau_\ell - 1]$. The update criterion is given in line 2, where the next starting slot for ℓ, s_ℓ^{n+1}, is determined such that the new block $[s_\ell^{n+1} : s_\ell^{n+1} + \tau_\ell - 1]$ does not overlap with current bookings $T_{\ell'}(s_n^\ell)$ of interfering links $\ell' \in I(\ell)$. Fig. 2(c) illustrates the new slot booking at the update step, where $I(\ell) = \{l_i\}_{i=0}^4$. This figure is discussed further below as part of implementation.

A. Distributed Implementation

Note that during each update, say n-th update for ℓ, the starting point s_ℓ^{n+1} for the next block is determined only based

[1]We note that this scaling factor τ_s may not necessarily be τ from the previous section.

This article has been accepted for publication in IEEE/ACM Transactions on Networking. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TNET.2023.3273349

Algorithm 1 Static Slot Reservation Algorithm

1 – Initial bookings

Consider any arbitrary ordering of links in $\mathcal{L}$ as $\{l_i\}_{i=1}^{|\mathcal{L}|}$.
$s_{l_1}^1 = 1$, and $T_{l_1}(1) := [s_{l_1}^1 : s_{l_1}^1 + \tau_{l_1} - 1]$
$s_{l_i}^1 = 0$ for each $i = 2, \ldots, |\mathcal{L}|$.
for $i = 2$ to $|\mathcal{L}|$ **do**
  $I_0(l_i) = \{l_j\}_{j=1}^{i-1} \bigcap I(l_i)$. // These are links in $I(l_i)$ for which initial allocation has been computed in previous iterations.
  $s_{l_i}^1 := \inf\{k \in \mathbb{N} : [k : k+\tau_{l_i}-1] \bigcap \bigcup_{l \in I_0(l_i)} T_l(1) = \phi\}$, // A non-conflicted block of slots for l_i is chosen for initialization.
  $T_{l_i}(1) := [s_{l_i}^1 : s_{l_i}^1 + \tau_{l_i} - 1]$.
end for

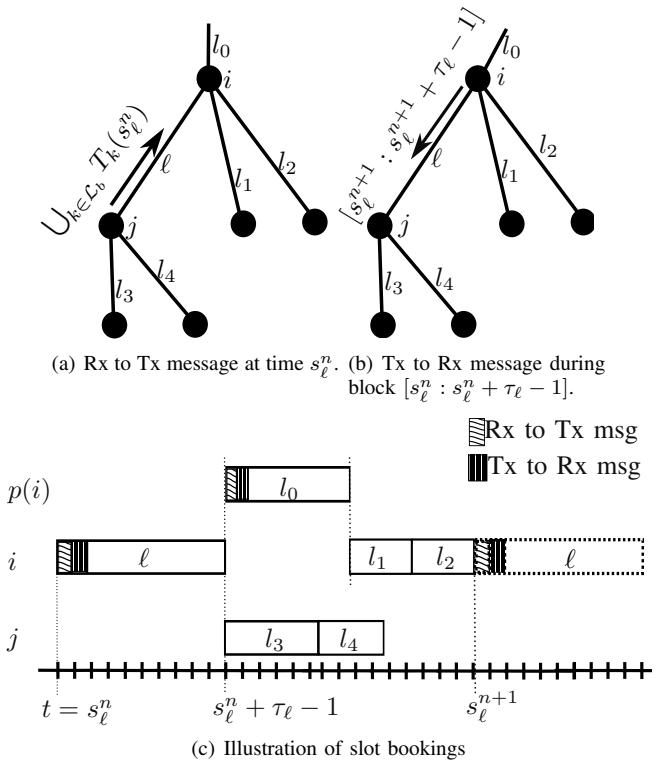
2 – Update Rule which generates subsequent bookings

$t = 1$.
$n_i = 1$ for each $i = 1, \ldots, |\mathcal{L}|$.
while $t \geq 1$ **do**
  for $i = 1, \ldots, |\mathcal{L}|$ **do**
    if $t = s_{l_i}^{n_i} + \tau_{l_i} - 1$ **then**
      $s_{l_i}^{n_i+1} := \inf\{k \in \mathbb{N} : \ k > t, [k : k + \tau_{l_i} - 1] \bigcap \bigcup_{l \in I(l_i)} T_l(t) = \phi\}$ // The next block of allocation for link l_i is decided here. The algorithm searches for the earliest non-conflicted block of slots for link l_i starting from slot $t+1$.
      $T_{l_i}(t+1) := [s_{l_i}^{n_i+1} : s_{l_i}^{n_i+1} + \tau_{l_i} - 1]$
      $n_i \leftarrow n_i + 1$
    else
      $T_{l_i}(t+1) := T_{l_i}(t) - \{t\}$
    end if
  end for
  $t \leftarrow t + 1$
end while

on the local information of links in $I(\ell)$. In the following, we present a local message passing scheme, which ensures that each link ℓ has the necessary information $\bigcup_{\ell' \in I(\ell)} T_{\ell'}(t)$ from links in $I(\ell)$ during its update at t. An illustration of each stage of the local message passing scheme is shown in Fig. 2.

In the figure, the link $\ell = (i, j)$, is beginning its transmission of block $[s_\ell^n : s_\ell^n + \tau_\ell - 1]$ at time s_ℓ^n. During this time interval, the transmitter gNB i, will book link ℓ's next time-block $[s_\ell^{n+1} : s_\ell^{n+1} + \tau_\ell - 1]$ and pass this information to node j, as the Tx to Rx message, shown in Fig. 2(b). Note that none of the neighbouring links $\{l_a\}_{a=0}^4$ will be transmitting during this time-block, so these neighbouring links will not be booking their next time-blocks during the interval $[s_\ell^n : s_\ell^n + \tau_\ell - 1]$.

Since node j is a gNB, it needs to know about link ℓ's newly booked time-block for the future bookings of its own downstream links l_3, l_4. Consider any downstream link $\ell' \in \mathcal{L}_j = \{l_3, l_4\}$. At time s_ℓ^n, link ℓ''s next time-block has already been booked, although it is still in the future, *e.g.* in Fig. 2(c), the next time-block of l_3 (or l_4) is $[s_\ell^n + \tau_\ell : s_\ell^n + \tau_\ell + \tau_{l_3} - 1]$ (or $[s_\ell^n + \tau_\ell + \tau_{l_3} : s_\ell^n + \tau_\ell + \tau_{l_3} + \tau_{l_4} - 1]$ respectively). When this time-block (which is *after* time $s_\ell^n + \tau_\ell - 1$) starts, node j will therefore already have the necessary information about link ℓ, due to the Tx to Rx msg from i (see again Fig. 2(b)). For the same reason, node i already has the booking information for the link l_0 from its parent $p(i)$, which it received at the earlier time when it was last acting as a receiver. Node i also knows the booked times of its other downstream links, since it is the transmitter. Hence, i has the booking information $\bigcup_{a=0}^2 T_{l_a}(s_\ell^n)$ of all the links connected to it, at time s_ℓ^n, because of the Tx to Rx msgs in the message passing scheme.



(a) Rx to Tx message at time s_ℓ^n. (b) Tx to Rx message during block $[s_\ell^n : s_\ell^n + \tau_\ell - 1]$.

(c) Illustration of slot bookings

Fig. 2. Message passing implementation

Hence, to book the time-block $[s_l^{n+1} : s_l^{n+1} + \tau_l - 1]$ at time s_l^n, i only requires the information $\bigcup_{a=3}^4 T_{l_a}(s_\ell^n)$, *i.e.* the booking information of downstream links from j, which are the only remaining links in $I(\ell)$. In the message passing scheme, this information is sent from j to i in a short uplink transmission (Rx to Tx msg) in a mini slot at time s_ℓ^n, at the very start of the block. This is illustrated in Fig. 2(a). After this message, node i books the next time block, and this new booking information is sent from i to j, as the prior discussed Tx to Rx msg, during the transmission block $[s_\ell^n : s_\ell^n + \tau_\ell - 1]$, as illustrated in Fig. 2(b).

V. Convergence Results

In this section, we present our main results regarding the convergence of the SSR algorithm.

We define *window size*, $w(t)$, as the smallest positive integer such that each link $\ell \in \mathcal{L}$ has at least τ_ℓ allocated slots in the interval $[t - w(t) + 1 : t]$. For $t \geq \max_{\ell \in \mathcal{L}} s_\ell^1 + \tau_\ell - 1$,

$$w(t) := \min\{w \in \mathbb{N} : \sum_{t'=t-w+1}^{t} U_\ell(t') \geq \tau_\ell, \forall \ell \in \mathcal{L}\}, \quad (6)$$

It is clear that $w(t) \geq f^* \tau_s$, which is the minimum possible value due to LP (3). First, we show that the window size $w(t)$ is monotonically non-increasing with t, in Theorem 2. Next, we prove convergence of $w(t)$ to the optimal value $f^* \tau_s$ in

Theorem 3, by showing $\exists T \leq \max_{\ell\in\mathcal{L}}(s^1_\ell + \tau_\ell) + \sum_{\ell\in\mathcal{L}} \tau_\ell$, such that $w(T) = f^*\tau_s$. Clearly, the convergence occurs within a bounded time linear in the size of the network.

The convergent rates under the SSR algorithm achieve the given rate vector $\boldsymbol{\alpha}$, since for any time $t > T$, each link $\ell \in \mathcal{L}$ receives τ_ℓ slots in the interval $[t : t + f^*\tau_s - 1]$.

Theorem 2. *Under Algorithm 1, the window size is non-increasing. For* $t \geq \max_{\ell\in\mathcal{L}} s^1_\ell + \tau_\ell - 1$,

$$w(t+1) \leq w(t). \tag{7}$$

Proof. See Appendix A. □

Theorem 3. *The window-size* $w(t)$ *for the SSR algorithm, Algorithm 1, converges to* $f^*\tau_s$ *before slot* $T' := \max_{\ell\in\mathcal{L}}(s^1_\ell + \tau_\ell) + \sum_{\ell\in\mathcal{L}} \tau_\ell$*, where* f^* *is the optimal value in the LP* (3)*,* i.e. $\exists T \leq T'$ *such that* $w(t) = f^*\tau_s$, $\forall t \geq T$. *Furthermore,* $f^* = \max_{b\in\mathcal{B}} \gamma_{\ell_b} + \sum_{\ell\in\mathcal{L}_b} \gamma_\ell$. *Here, we take* $\gamma_{\ell_r} = 0$ *since root* r *has no parent node.*

Proof. See Appendix B. □

Note that the allocation in any post-convergence interval $[t : t + f^*\tau_s - 1]$, yields an optimal solution $\{f^*_S\}_{S\in\mathcal{S}}$ of LP (3), as follows.

$$f^*_S = \frac{1}{\tau_s} \sum_{t'=t}^{t+f^*\tau_s-1} \mathbb{I}(S = \{\ell \in \mathcal{L} : U_\ell(t') = 1\}) \tag{8}$$

for each $S \in \mathcal{S}$, where $\mathbb{I}(\cdot)$ is the indicator function.

Note that the SSR algorithm converges to the optimal solution of (3) irrespective of the achievability of the traffic rate vector $\boldsymbol{\alpha} \in \mathbb{Q}^{|\mathcal{L}|}$, due to Theorem 3. The rate for user $u \in \mathcal{U}$ that the SSR converges to is at least α_u/f^*, for each $u \in \mathcal{U}$. If $f^* < 1$ then $\boldsymbol{\alpha}$ is an achievable rate vector. If $f^* > 1$ then $\boldsymbol{\alpha}/f$ is only an achievable rate vector for any $f > f^*$.

At this point, we have presented our SSR algorithm and discussed its implementation of slot bookings using message passing. We have also shown that window size monotonically converges to the optimal value in a number of slots that is linear in the size of the network. As such, it achieves any traffic rate vector in the achievable rate region.

VI. Dynamic slot reservation algorithm

The SSR algorithm is applicable for a static setup, where the UE requests are fixed. When the flow requests are dynamic, straightforward implementation of the SSR requires re-initialization every time the network changes state. In this section, we consider a dynamic model, with flows that arrive at random times, and which depart after their data demands have been met. We propose a dynamic slot reservation (DSR) algorithm, which dynamically adapts the flow rates and allocates slots to the flows using slot reservation, in a similar way to the SSR algorithm, but without re-initialization.

In the dynamic scenario, we assume that UE file requests arrive at a gNB $b \in \mathcal{B}$, as an exogenous process. The size of an arriving UE file is i.i.d exponential with mean D bits. The UE file request departs the network once the file is fully downloaded. For a UE request u at gNB b, the access rate $R_{(b,u)}$ (in bits/slot), takes a value from the set $\{R^a_1, \ldots, R^a_K\}$, which is fixed until u departs[2]. We call the UE requests with access rate R^a_k, as the k-th class for $k = 1, \ldots, K$. The arrival process for k-th class of UE requests at gNB b forms a Poisson process with rate $\nu^{(k)}_b$ requests per slot.

The network graph in the dynamic case changes with UE file request arrivals and departures. With a new arrival of UE request u at gNB b, a new node u and an access link (b, u) is added to the graph. Similarly, a node and an access link are removed from the graph with each UE request departure. The traffic of an active UE request u at a gNB $b \in \mathcal{B} - \{r\}$ has to be routed from root r to node u along the path connecting r to b. We refer to this traffic flow as being associated with each link along this path. As before, R_{ℓ_b} (in bits/slot) is the fixed rate of the backhaul link ℓ_b, connecting a parent gNB $p(b)$ and its child gNB b.

Let $N^{(k)}_b(t)$ denote the number of UE requests of k-th class at a gNB $b \in \mathcal{B}$, and $\mathcal{N}(t) := \{N^{(k)}_b(t)\}_{b\in\mathcal{B},k\in[1:K]}$ denote the network state, in slot t.

In the following Algorithm 3, we present our Dynamic Slot Reservation (DSR) Algorithm, which adapts the τ_ℓ values (as part of flow rate control) based on the state $\mathcal{N}(t)$. We now briefly discuss the criteria for deciding τ_ℓ values and present results.

A. Stationary flow control policy

Consider a flow control policy which provides an equal end-to-end rate $\alpha_\mathcal{N}$ bits/slot for each flow in the network in state $\mathcal{N}$. Under this policy, the utilization γ_ℓ of a link ℓ is given by $\gamma_\ell = \alpha_\mathcal{N} N_\ell / R_\ell$, where N_ℓ is the number of flows using link ℓ. It follows from Theorem 3, that for feasibility, $\gamma_{\ell_b} + \sum_{\ell\in\mathcal{L}_b} \gamma_\ell \leq 1$ for each $b \in \mathcal{B}$. Hence for the flow rates to be feasible, $\alpha_\mathcal{N} \leq 1/F_b(\mathcal{N})$ for each $b \in \mathcal{B}$, where

$$F_b(\mathcal{N}) := \frac{N_{\ell_b}}{R_{\ell_b}} + \sum_{k=1}^{K} \frac{N^{(k)}_b}{R^a_k} + \sum_{b':p(b')=b} \frac{N_{\ell_{b'}}}{R_{\ell_{b'}}} \tag{9}$$

Hence, $(\max_{b\in\mathcal{B}} F_b(\mathcal{N}))^{-1}$ is the maximum value for flow rate $\alpha_\mathcal{N}$, *i.e.* it is not possible to provide a higher rate for all flows. Our policy for setting $\boldsymbol{\tau}$, as used in Algorithm 1, based on the state $\mathcal{N}$, achieves a rate of at least $(\max_{b\in\mathcal{B}} F_b(\mathcal{N}))^{-1}$ bits/slot for each flow after Algorithm 1 has converged, assuming no change in the state $\mathcal{N}$.

Our following policy (10), (11), determines the vector $\boldsymbol{\tau}$ for any given state $\mathcal{N} \in \mathbb{Z}_+^{|\mathcal{B}|\times K}$:

$$\tau_{(b,u)} = \frac{\tau_s \times 1 \text{ bit/slot}}{R_{(b,u)}} \tag{10}$$

for each active UE request u at gNB $b \in \mathcal{B}$, and

$$\tau_{\ell_b} = \tau_s \frac{N_{\ell_b} \times 1 \text{ bit/slot}}{R_{\ell_b}} \tag{11}$$

[2]In this section, we assume the link rate to be fixed on the time-scale of flow durations. In Section VII we consider a model in which the link rate changes due to transitions between a LoS and a NLoS state. We also consider access link rate changes in Section IX-C, in the context of rate guarantees. Section X provides possible directions for considering full blockage, and network topology changes in future work.

This article has been accepted for publication in IEEE/ACM Transactions on Networking. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TNET.2023.3273349

for backhaul link ℓ_b, for $b \in \mathcal{B}$, where N_{ℓ_b} is the total number of active flows which are using ℓ_b and τ_s is the scaling factor.

Recall that each access link ℓ reserves a block of τ_ℓ slots per window in Algorithm 1. Therefore, under (10), an access link (b, u) is able to transmit τ_s bits in a window. Similarly, a backhaul link ℓ_b transmits τ_s bits in a window for each flow using ℓ_b, which yields (11). We show in Theorem 4 that under this proposed scheme, each flow transmits at an end-to-end rate of τ_s bits per window.[3]

Theorem 4. *For a fixed state $\mathcal{N} \in \mathbb{Z}_+^{|\mathcal{B}|\times K}$, implementing Algorithm 1 (SSR) with the τ_ℓ values from (10), (11), leads to the following steady state conditions.*

1) *The converged window size is given by $\tau_s \max_{b\in\mathcal{B}} F_b(\mathcal{N})$ slots, where $F_b(\mathcal{N})$ is given in (9).*
2) *Each flow receives a flow-rate of at least $(\max_{b\in\mathcal{B}} F_b(\mathcal{N}))^{-1}$ bits/slot.*

Proof. It follows from Theorem 3 that the converged window size equals $\max_{b\in\mathcal{B}} \tau_{\ell_b} + \sum_{\ell\in\mathcal{L}_b^{\mathcal{N}}} \tau_\ell$, where $\mathcal{L}_b^{\mathcal{N}}$ is the set of downstream links from node b when the network is in state $\mathcal{N}$. The first result is now immediate from (10) and (11).

Under the SSR algorithm with (11), a backhaul link ℓ_b is allocated at least $\tau_s N_{\ell_b}/R_{\ell_b}$ slots in a window, which implies each flow through ℓ_b is allocated at least τ_s/R_{ℓ_b} slots per window. Hence, there is a net transfer of at least τ_s bits for each flow passing through ℓ_b in a window. Similarly, for a access link (b, u), there are at least $\tau_s/R_{(b,u)}$ allocated slots in a window. Hence, there is a net transfer of at least τ_s bits for each UE request u in a window. Hence, there is a net end-to-end flow rate of at least τ_s bits per window for each UE request in the network. The second result now immediate from the first result on window-size. □

The goal of the DSR algorithm is to implement the SSR algorithm, Algorithm 1, while adapting the τ_l values according to (10), (11). In Section VIII, we use Theorem 4 to prove stability for a policy which instantaneously realizes the steady state flow rates of the DSR algorithm (given in Theorem 4) whenever the network state changes .

For the DSR algorithm, we need to address a few practical considerations which occur in the dynamic case. We discuss these and describe the DSR algorithm in the following section.

B. The Dynamic Slot Reservation (DSR) Algorithm

The DSR algorithm implements the flow control policy in (10), (11) along with Algorithm 1, with modifications to address the following practical considerations.

1) *Change of state:* When the state $\mathcal{N}$ changes due to an arrival or a departure, the τ_ℓ values for the existing links also change. The straightforward approach is to re-initialize and run the slot reservation algorithm, Algorithm 1, with the new τ_ℓ values. However, such reconfiguration is costly in terms of message communication across the network.

In the DSR algorithm, suppose the τ_ℓ value changes due to a change of state at slot t. The link ℓ will simply book its next block of slots according to the updated τ_ℓ value, after its current ending slot. Hence, no reconfiguration is needed for state changes under the DSR algorithm[4].

2) *Small queue sizes:* By choosing τ_ℓ according (10), (11), each link reserves enough slots to transmit τ_s bits per flow, for each flow using the link in a window. This will lead to wasted slots when the queue size corresponding to a flow is smaller than τ_s, *e.g.* when a flow is about to depart and has only a few residual bits of traffic left.

In the DSR algorithm, we modify the τ_ℓ calculation to address this issue. We avoid booking the extra slots, by considering the minimum of τ_s and q_n^u, where q_n^u is the bits corresponding to UE request u, at gNB n in lines 3 and 6 in the LOAD UPDATE procedure in Algorithm 2.

3) *Scaling factor:* Thus far, the scaling factor τ_s is chosen such that τ_ℓ is an integer for each ℓ. In DSR algorithm, we do not impose this restriction, instead we round up the τ_ℓ value to the nearest integer in lines 3 and 6 in the LOAD UPDATE procedure in Algorithm 2.

4) *Reservation for new links:* As mentioned earlier, the arrival of a flow adds a new access link to the graph in the dynamic case. The DSR algorithm has to allocate the initial block for the newly added link, after which subsequent allocations can occur by the updating at the end of block. The initial block has to be allocated such that it does not conflict with the existing allocations.

In the DSR algorithm, the initial allocation of a new access link at gNB n is performed in the same block as the earliest updating access link at gNB n (see lines 26-28 of RESERVATION UPDATE procedure in Algorithm 2).

In the special case when there are no existing access links at a gNB n, we reserve one slot which does not conflict with the backhaul links connected to n (see line 23 in RESERVATION UPDATE procedure in Algorithm 2)[5]. This slot will be used to reserve the initial block for future access link arrivals at n. This case is denoted by the FUTURE argument for ℓ in the RESERVATION update procedure in Algorithm 2).

The procedures LOAD UPDATE, NEW ARRIVAL RESERVATION and RESERVATION UPDATE which are executed at each gNB n are presented in Algorithm 2. As mentioned, the LOAD UPDATE is a sub-procedure used to compute the τ_ℓ values at each block reservation. The NEW ARRIVAL RESERVATION is a sub-procedure used to reserve the first block for new access links, *i.e.* UE request arrivals. RESERVATION UPDATE is the main procedure which calls the other two procedures. RESERVATION UPDATE books the initial reservations for new links and updates the reservations for existing links. It also reserves one slot (for future access links) in the special case when there are no existing access links.

The main DSR algorithm is presented in Algorithm 3. The lines 1-8 are initialization, where one slot is reserved for each

[3]We remark that this is not the only way one could assign values to the τ_ℓ's. For example, one can construct weighted assignments which give priorities to different classes of UEs, or to flows with different numbers of hops.

[4]Note that the same approach of adapting τ_l's can be used to deal with changing access link rates, *e.g.* due to changes in LoS/NLoS state of the link in high mobility scenarios. We take this approach in the numerical results in Section VII.

[5]We remark that this does not lead to capacity wastage for the following reasons. 1) The fraction of window used by this one slot is $\frac{1}{w(t)}$ in the worst case, and $w(t)$ keeps increasing as the number of flow increases. 2) Once a new flow arrives at the gNB, this extra slot is not booked.

This article has been accepted for publication in IEEE/ACM Transactions on Networking. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TNET.2023.3273349

backhaul link in the network, and one special slot for each gNB for FUTURE links. $\mathcal{U}_n^{cur}$ is set of existing UE requests at gNB n. We note that $\{(n,u) : u \in \mathcal{U}_n^{cur}\}$ is the set of access links of n. $\mathcal{U}_n^{new}$ is the set of new UE requests, awaiting their initial reservation. Once the initial reservation is completed, a UE request is moved from $\mathcal{U}_n^{new}$ to $\mathcal{U}_n^{cur}$ in lines 14-15 in NEW ARRIVAL RESERVATION procedure. Naturally, a UE request arrival is added to $\mathcal{U}_n^{new}$ in line 18 and a departure is removed from $\mathcal{U}_n^{cur}$ in line 19.

Lines 12-16 show the update process in the DSR algorithm, which happens during the block just like in Algorithm 1.

Algorithm 2 Procedures for DSR Algorithm

1: **procedure** LOAD UPDATE(n,ℓ) // n is the transmitter of link ℓ.
2: **if** ℓ is a backhaul link **then**
3: $\tau_\ell := \lceil \frac{1}{R_\ell} \sum_{u:\ell \in P_u} \min(q_n^u, \tau_s) \rceil$
4: $\tau_\ell \leftarrow \max(1, \tau_\ell)$
5: **else if** ℓ is an access link **then**
6: $\tau_\ell := \lceil \frac{\min(\tau_s, q_n^u)}{R_\ell} \rceil$, where u is the receiver of ℓ.
7: **end if**
8: **end procedure**
9: **procedure** NEW ARRIVAL RESERVATION(n)
10: **for** $u \in \mathcal{U}_n^{new}$ **do**
11: **if** $q_n^u > 0$ **then**
12: Run LOAD UPDATE(n,(n,u)) to compute $\tau_{(n,u)}$.
13: Reserve $\tau_{(n,u)}$ slots for link (n,u) using the update rule of Algorithm 1.
14: $\mathcal{U}_n^{new} \leftarrow \mathcal{U}_n^{new} - \{u\}$.
15: $\mathcal{U}_n^{cur} \leftarrow \mathcal{U}_n^{cur} \bigcup \{u\}$.
16: **end if**
17: **end for**
18: **end procedure**
19: **procedure** RESERVATION UPDATE(n,ℓ) // Here, either ℓ is a link of gNB n, or in the special case with no UEs are at gNB n, argument ℓ can be FUTURE, where a single slot will be reserved which will be used for reserving slots for future UE request arrivals at gNB n.
20: **if** ℓ is FUTURE **then**
21: Run NEW ARRIVAL RESERVATION(n).
22: **if** $\mathcal{U}_n^{cur} = \phi$ **then**
23: Reserve one slot, which does not conflict with backhaul links $\{\ell_n\} \bigcup_{m \in \mathcal{B}: p(m)=n} \{\ell_m\}$. // This slot will be used to reserve slots for future UE arrivals at n.
24: **end if**
25: **else if** ℓ is an access link **then**
26: Run LOAD UPDATE(n,ℓ).
27: Reserve τ_ℓ slots for link ℓ according to the update rule of Algorithm 1.
28: Run NEW ARRIVAL RESERVATION(n).
29: **else if** ℓ is a backhaul link **then**
30: Run LOAD UPDATE(n,ℓ).
31: Reserve τ_ℓ slots for ℓ according to the update rule of Algorithm 1.
32: **end if**
33: **end procedure**

Algorithm 3 Dynamic Slot Reservation (DSR) Algorithm

1: $U_n^{cur} = \phi, U_n^{new} = \phi$ for each $n \in \mathcal{B}$. // Initialization
2: Consider any arbitrary ordering of gNBs in $\mathcal{B}$ as $\{b_i\}_{i=1}^B$.
3: **for** i = 1 to B **do**
4: **if** $b_i \neq r$ **then**
5: Run RESERVATION UPDATE($p(b_i)$,ℓ_{b_i}).
6: **end if**
7: Run RESERVATION UPDATE(b_i,FUTURE).
8: **end for**
9: $t = 1$.
10: **while** $t \geq 1$ **do** // Main Algorithm
11: **for** $n \in \mathcal{B}$ **do**
12: **if** t is the ending slot of a link ℓ of gNB n. **then**
13: Run RESERVATION UPDATE(n,ℓ).
14: **else if** t is a slot reserved for FUTURE links **then**
15: Run RESERVATION UPDATE(n,FUTURE).
16: **end if**
17: Let $A_n(t)$ be the set of new UE requests which arrived during slot t, and $D_n(t)$ be the set of the UE requests which departed during slot t.
18: $\mathcal{U}_n^{new} \leftarrow \mathcal{U}_n^{new} \bigcup A_n(t)$.
19: $\mathcal{U}_n^{cur} \leftarrow \mathcal{U}_n^{cur} - D_n(t)$.
20: **end for**
21: $t \leftarrow t + 1$.
22: **end while**

VII. NUMERICAL RESULTS

In this section, we compare the performance of the proposed DSR algorithm with a NUM based congestion control algorithm, joint-MWM algorithm from [1], [2]. In each slot, the joint-MWM algorithm updates the flow rates based on the link costs in that slot. In each slot, the max-weight schedule of links is computed and used for transmission. The link costs are treated as queue sizes and are updated according to a queuing rule. The flow rates under the joint-MWM algorithm was shown in [1] to converge to the NUM solution for a fixed set of flows .

The implementation of joint-MWM algorithm involves finding the max-weight schedule in *each slot*. In contrast, our algorithm only requires local message passing, *only once per window* for most links. As mentioned in the introduction, finding the max-weight schedule involves message passing to the root and back. Hence for joint-MWM, the computational complexity is linear in the number of links, and the message passing overhead is linear in the number of gNBs [6]. In contrast for DSR, computational complexity (which is for slot bookings), and the message passing overhead at a gNB, is linear with the number of neighbors (since a gNB only communicates with its neighboring gNBs).

We consider the IAB setup shown in Figure. 3. Here, gNB 1 is the IAB donor, and the other gNBs $2-5$ are IAB nodes with the IAB topology shown in Figure. 3. The parameters for simulation are chosen as follows. For the gNB-gNB backhaul links, the distance is uniformly chosen between 100 m and 600 m. For the randomly chosen realization, the backhaul link rate vector $[R_{\ell_i}]_{i=2}^5$ is $[13.19, 8.92, 11.83, 12.58]$ Gbps.

This article has been accepted for publication in IEEE/ACM Transactions on Networking. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TNET.2023.3273349

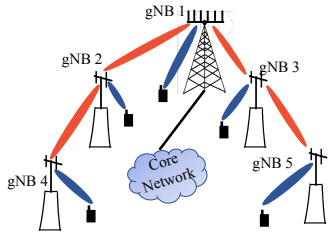


Fig. 3. Simulated IAB Network Topology

At each gNB, the flow (UE request) arrival process is an i.i.d Poisson process with rate ν (in requests/sec). The size of each arriving flow request is i.i.d exponential with mean 50 Mb (rounded up to an integral number of packets). The gNB-UE distance for each arriving UE request is chosen uniformly randomly between 0 and 300m. Other parameters are given in the following Table I.

We consider two scenarios. In the first scenario, Sc. 1, each access link is in a LoS state independently with probability 0.9, and in a NLoS state otherwise. In a given state, the path-loss is decided independently according to the 3GPP Urban Micro channel model. Following [25], we consider Rician fading for access links with K factor as 13 dB for LoS links and 6 dB for NLoS links. The access link rate is taken to be the Shannon rate and is *fixed* until the flow's departure.

For the second scenario, Sc. 2, we model the rate for an access link as a time varying process as follows. The time spent in the LoS (and the NLoS) state is distributed as an *i.i.d* geometric random variable with mean 500 (and $500/9$) slots respectively. The mean length of time in the LoS state is therefore 62.5 msec, and the mean time in the NLoS state is 6.9 msec (from the parameters in Table I). The stationary probability of an access link being in LoS state is 0.9. After each state change, the access link rate is realized independently, as described in the first scenario. Thus, the channel alternates between a good (LoS) and bad (NLoS) state, similar to a Gilbert-Elliot process, but with a randomly selected rate due to Rician fading.

The DSR algorithm was designed for the case that link rates of flows do not change, as in Sc. 1. If UEs are pedestrians, the state changes between LoS and NLoS will be on the order of seconds, during which time most flows will complete. Sc. 1 is therefore applicable when UEs are moving at pedestrian speeds. The rate of change of link state is much higher in Sc. 2, so Sc. 2 is applicable to UEs moving at higher speeds.

In Sc. 2, we extend the DSR algorithm enabling it to adapt to time varying access link rates. During a load update for an access link l, the new τ_l is calculated using the access-link rate in the slot in which the booking is made. If the access link rate changes during a transmission block, the number of bits transmitted changes accordingly. At the next load update, the new τ_l will be calculated using the access-link rate at the time of the new booking, and so on.

A. Results

End-to-end delay for a flow is the amount of time from its arrival till its departure. We compare the average end-to-end delays for multi-hop flows under the DSR and joint-MWM algorithms for various arrival rates ν.

TABLE I
SIMULATION PARAMETERS

Parameter	Value
Carrier frequency	23 GHz
Bandwidth	1 GHz
Propagation model	3GPP Urban Micro
Slot duration	125 μs
Packet size	100 Kb
τ_s	200
Noise spectral density	-174 dBm/Hz
gNB transmit power	30 dBm
Beamforming gain	30 dB (for access), 40 dB (for backhaul)
Noise figure	5 dB (for gNB), 7 dB (for UE)

Fig. 4 presents the results for 1-hop flows, *i.e.* flows for UEs connected to gNB 1. The main observation from Fig. 4 is that Joint-MWM has the lowest delays for these flows.

Observe also that channel variation (*i.e.* Sc. 2) *decreases* the end-to-end delay for both DSR and Joint-MWM. *Multi-user diversity* is a factor behind the delay improvement for joint-MWM, since scheduling decisions are made every slot.[6] Channel variation also decreases the delays for the DSR algorithm. In the DSR algorithm, more slots are reserved by links with smaller rates, and hence a state change from NLoS to LoS (which occurs in Sc. 2) reduces resource usage.

The benefits of channel variation are also apparent in Figs. 5-6 below for all the schemes.

Fig. 5 presents the results for 2-hop flows, *i.e.* flows for UEs connected to gNB 2 or 3. In Sc. 1, DSR has lower delay than Joint-MWM for lower arrival rates, and the trend reverses for high arrival rates. The same phenomenon occurs in Sc. 2.

Fig. 6 presents the results for 3-hop flows, *i.e.* flows of UEs connected to gNBs 4 or 5. Here, joint-MWM has the highest delays in both scenarios. It can be observed that Joint-MWM favours flows with less hops, in terms of end-to-end delay. In contrast, the DSR algorithm provides a more uniform end-to-end delay across the flows.

In Fig. 7, we directly compare end-to-end delays of 1-hop, 2-hop and 3-hop flows when the arrival rate is moderate ($\nu = 29.5$) and when the arrival rate is high ($\nu = 38.5$). This is done for all schemes in both Scenarios. Fig. 7 shows that the uniform flow rate allocation under DSR leads to more uniform delays across all flows (from 1-hop flows to 3-hop flows), as compared to Joint-MWM. This feature of DSR makes it very well suited for IAB networks.

The DSR algorithm exhibits similar behaviour in terms of achieved stability region, (*i.e.* arrival rates such that the expected delays are bounded), as the joint-MWM algorithm. In [1], [2], Joint-MWM is was shown to be stable (*i.e.* leads to bounded expected delays) for all feasible arrival rates for the case of fixed link rates (*e.g.* Sc. 1). We do not have a proof that the DSR algorithm is similarly always stable in Sc. 1, nor for any of the algorithms when the link rates are time varying. However in the following section, we will prove the stability

[6]This factor is also one reason (amongst several) as to why such an algorithm is inherently impractical across a multi-hop network, especially when the channel variation is fast.

This article has been accepted for publication in IEEE/ACM Transactions on Networking. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TNET.2023.3273349

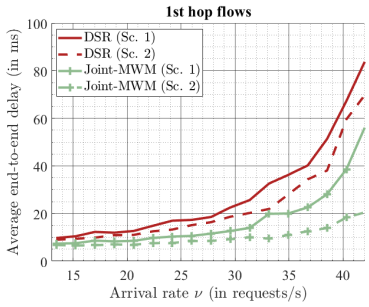


Fig. 4. Average end-to-end delays for flows accessing via gNB 1

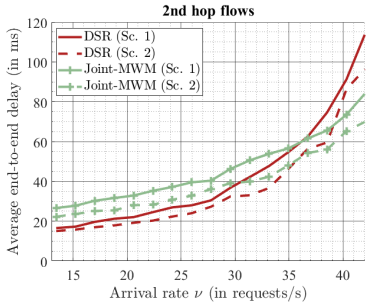


Fig. 5. Average end-to-end delays for flows accessing via gNBs 2 or 3

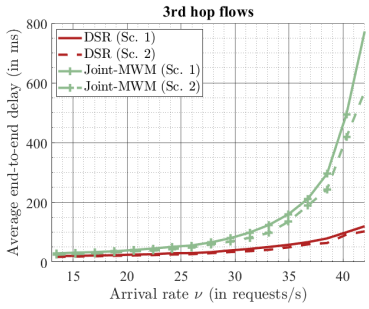
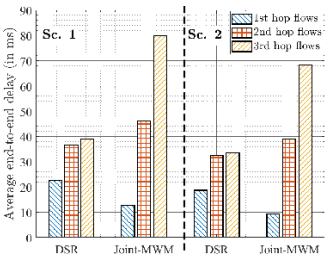


Fig. 6. Average end-to-end delays for flows accessing via gNBs 4 or 5

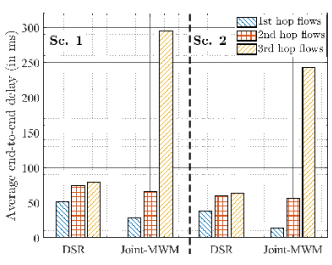
of a continuous-time abstraction of the DSR algorithm for the case of fixed access link rates.

VIII. Stability of Dynamic Flow Control

In this section, we will show the stability of a continuous-time abstraction of the DSR algorithm. Under the continuous-time abstraction, we assume that each flow instantaneously receives a rate of $(\max_{b\in\mathcal{B}} F_b(\mathcal{N}))^{-1}$ bits/slot when the network enters state $\mathcal{N}$, *i.e.* we ignore the transient period required for convergence of slot reservation. Secondly, we assume that the state changes happen in continuous-time, *i.e.* not just at the slot boundaries. In this section, a slot is simply an arbitrary unit of time. We note that the assumptions of continuous time abstraction are justified under a time-scale separation between the slot sizes and flow durations, *i.e.* when the file sizes are large and the slot sizes are small.



(a) $\nu = 29.5$ requests/s



(b) $\nu = 38.5$ requests/s

Fig. 7. Comparison of delays across flows, as well as across schemes.

In this section, we consider a continuous time model for state process $\{\mathcal{N}(t_c)\}_{t_c\in\mathbb{R}_+}$. Here, $t_c \in \mathbb{R}_+$ represents the continuous time. $N_b^{(k)}(t_c)$ denote the number of UE requests of k-th class at a gNB $b \in \mathcal{B}$ at time t_c, and $\mathcal{N}(t_c) := [N_b^{(k)}(t_c)]_{b\in\mathcal{B},k\in[1,K]}$ denote the network state, at time t_c.

First, we define what we mean by stationary flow control policies in Section VIII-A and characterize the stability region. We then propose a particular flow control policy in Section VIII-B, which is intended as an abstraction of the discrete-time DSR algorithm, where the flow rates are set by (10),(11), and determined by the window sizes given in Theorem 4, but here such rates are realized instantaneously. We will show that this flow control algorithm achieves every vector in the stability region.

A. Stationary Flow Control Policies

Fixing a network state $\mathcal{N} \in \mathbb{Z}_+^{|\mathcal{B}|\times K}$, let $\mathcal{G}^{\mathcal{N}}$ denote the network graph, $\mathcal{U}^{\mathcal{N}}$ the set of UE requests, and $\mathcal{L}^{\mathcal{N}}$ be the set of links in $\mathcal{G}^{\mathcal{N}}$, when the network is in the state $\mathcal{N}$.

A *stationary* flow control policy decides the flow rate vector $\boldsymbol{\alpha}^{\mathcal{N}} := [\alpha_u^{\mathcal{N}}]_{u\in\mathcal{U}^{\mathcal{N}}}$ when the network is in state $\mathcal{N}$. Let $\boldsymbol{\gamma}^{\mathcal{N}} := [\gamma_l^{\mathcal{N}}]_{l\in\mathcal{L}^{\mathcal{N}}}$ be the utilization vector corresponding to $\boldsymbol{\alpha}$, where $\gamma_l^{\mathcal{N}} := \sum_{u:l\in P_u} \alpha_u^{\mathcal{N}}/R_l$, and P_u is the set of links in the path from u to r. The minimum resource clearing problem for this setup can be formulated as LP (3) with $\mathcal{L}$ replaced by $\mathcal{L}^{\mathcal{N}}$. If the optimal value of the LP is less than or equal to 1, then it is possible to satisfy the flow rates $\boldsymbol{\alpha}^{\mathcal{N}}$ by time-sharing among the feasible sets of the network. Otherwise, from Theorem 1, $\boldsymbol{\alpha}^{\mathcal{N}}$ is infeasible.

From Theorem 3, the optimal value of the LP equals $\max_{b\in\mathcal{B}} \gamma_{\ell_b}^{\mathcal{N}} + \sum_{l\in\mathcal{L}_b^{\mathcal{N}}} \gamma_l^{\mathcal{N}}$, where $\mathcal{L}_b^{\mathcal{N}} \subseteq \mathcal{L}^{\mathcal{N}}$ is the set of downstream links of $b \in \mathcal{B}$. Hence, for feasibility, the flow rate vector $\boldsymbol{\alpha}^{\mathcal{N}}$ must satisfy the constraints (12)

$$\gamma_{\ell_b}^{\mathcal{N}} + \sum_{l\in\mathcal{L}_b^{\mathcal{N}}} \gamma_l^{\mathcal{N}} \le 1, \forall\ b \in \mathcal{B} \tag{12}$$

Let $\Lambda^{\mathcal{N}}$ be the set of all flow rate vectors $\boldsymbol{\alpha}^{\mathcal{N}}$ such that (12) holds.

Definition 4. *A* stationary *flow control policy θ is a mapping from $\mathbb{Z}_+^{|\mathcal{B}|\times K}$ to $\mathbb{R}_+^{\infty}$ such that*

$$\boldsymbol{\alpha}^{\mathcal{N}} = \theta(\mathcal{N}) \in \Lambda^{\mathcal{N}},\ \forall \mathcal{N} \in \mathbb{Z}_+^{|\mathcal{B}|\times K}. \tag{13}$$

i.e, each state $\mathcal{N} \in \mathbb{Z}_+^{|\mathcal{B}|\times K}$ is mapped to a feasible rate vector $\boldsymbol{\alpha}^{\mathcal{N}} \in \Lambda^{\mathcal{N}}$.

For the sake of brevity, we use the notation $(\cdot)(t_c)$ to mean $(\cdot)^{\mathcal{N}(t_c)}$ in the following. Hence, a stationary flow control

policy determines the flow rates $\boldsymbol{\alpha}(t_c)$, (and hence $\boldsymbol{\gamma}(t_c)$) at time t_c, based on the network state $\mathcal{N}(t_c)$, such that

$$\gamma_{l_b}(t_c) + \sum_{l \in \mathcal{L}_b(t_c)} \gamma_l(t_c) \leq 1, \forall\ b \in \mathcal{B} \tag{14}$$

As mentioned earlier, we assume that the flow rates set by the flow control policy are instantaneously realized by an underlying resource allocation policy which time-shares over the feasible sets, and hence the flow of each active UE request u is served at a rate of $\alpha_u(t_c)$ bits/sec. Note that the network state process $\{\mathcal{N}(t_c)\}_{t_c \in \mathbb{R}_+}$ is a Markov process, under a stationary flow control policy.

Recall that $\nu_b^{(k)}$ (requests/slot) is the rate of UE request arrivals of class k at gNB b. Let ν_{ℓ_b} be the aggregate rate of UE request arrivals, that use backhaul link ℓ_b, defined as

$$\nu_{\ell_b} := \sum_{b' \in \mathcal{B}-\{r\}} \mathbb{I}(\ell_b \in P_{b'}) \sum_{k=1}^{K} \nu_{b'}^{(k)} \tag{15}$$

where $P_{b'}$ is the set of links in the path from $b' \in \mathcal{B} - r$ to root r, and $\mathbb{I}(\cdot)$ is the indicator function.

Theorem 5. *The Markov process $\{\mathcal{N}(t_c)\}_{t_c \in \mathbb{R}_+}$ is not ergodic, under any stationary flow control policy, if*

$$D\left(\sum_{k=1}^{K} \frac{\nu_b^{(k)}}{R_k^a} + \sum_{b' \in \mathcal{B}: p(b')=b} \frac{\nu_{\ell_{b'}}}{R_{\ell_{b'}}} + \frac{\nu_{\ell_b}}{R_{\ell_b}}\right) > 1 \tag{16}$$

for some $b \in \mathcal{B}$

Proof. See Proof of Theorem 5 in Appendix C. □

B. Proposed flow control policy and stability

In this section, we consider the stability region of the flow control policy (first described in section VI-A) which provides a rate of at least $(\max_{b \in \mathcal{B}} F_b(\mathcal{N}))^{-1}$ bits/slot for each flow in the network, when in state $\mathcal{N}$.

Let $N_{\mathcal{B}}^{\max} := \max_{b \in \mathcal{B}} F_b(\mathcal{N})$. Under the continuous-time algorithm, each flow rate is set to be exactly $1/N_{\mathcal{B}}^{\max}$ bits per slot. The state evolution equations for this setup are given as

$$N_b^{(k)}(t_c) = N_b^{(k)}(0) + \mathcal{P}_{\nu_b^{(k)}}(t_c) - \mathcal{P}_{1/D}\left(\int_0^{t_c} \frac{N_b^{(k)}(s)}{N_{\mathcal{B}}^{\max}(s)} ds\right)$$

for $k = 1, \ldots, K$, where $\mathcal{P}_\lambda(.)$ represents the Poisson process with parameter λ.

The Markov process $\mathcal{N}(t_c)$ is ergodic, under the proposed flow control policy for every vector $\boldsymbol{\nu} := [\nu_b^{(k)}]_{b \in \mathcal{B}, k \in [1:K]}$ within the stability region, as shown in Theorem 6.

Theorem 6. *The Markov process $\{\mathcal{N}(t_c)\}_{t_c \in \mathbb{R}_+}$ is ergodic, under the proposed flow control policy, if*

$$D\left(\sum_{k=1}^{K} \frac{\nu_b^{(k)}}{R_i^a} + \sum_{b' \in \mathcal{B}: p(b')=b} \frac{\nu_{\ell_{b'}}}{R_{\ell_{b'}}} + \frac{\nu_{\ell_b}}{R_{\ell_b}}\right) < 1 \tag{17}$$

for each $b \in \mathcal{B}$

Proof. See Proof of Theorem 6 in Appendix C. □

IX. QoS GUARANTEES UNDER THE SLOT RESERVATION FRAMEWORK

In the previous section, we have shown how the DSR algorithm can be used to adapt flow rates and achieve stability. However, in applications with inelastic traffic, there can be stringent QoS requirements on flow rate, latency etc. In this section, we will provide a slot reservation framework to provide QoS guarantees, by proposing distributed admission control policies which can guarantee rate and latency.

The key idea is the following. As shown in Theorem 4 in section VI-A, the steady-state flow rate is a function of window-size. We use admission control to limit the window-size in order to meet QoS requirements.

We start by generalizing the criterion for choosing the load vector $\boldsymbol{\tau}$, which allows for different QoS requirements across flows. We generalize to a weighted version of τ_l's, which provides different rates for different classes $k = 1, \ldots, K$ of flows. A flow of class k has an access link rate R_k^a (as before) and a weight $w_k > 0$. For an access link of flow u of class k at a gNB $b \in \mathcal{B}$, we define

$$\tau_{(b,u)} = \frac{w_k}{R_k^a} \times \tau_s \text{ bits/slot} \tag{18}$$

and for a backhaul link, we define

$$\tau_{\ell_b} = \frac{\sum_{k=1}^{K} N_{\ell_b}^{(k)} w_k}{R_{\ell_b}} \times \tau_s \text{ bit/slot} \tag{19}$$

where, $N_{\ell_b}^{(k)}$ is the number of active flows of class k using link ℓ_b.

Using same arguments as in Theorem 4, we obtain the following equivalent Theorem about convergent window size and flow rates.

Theorem 7. *For a fixed state $\mathcal{N} \in \mathbb{Z}_+^{|\mathcal{B}| \times K}$, implementing Algorithm 1 with the τ_ℓ values from (18), (19), leads to the following steady state conditions.*

1) *The converged window size is given by $\tau_s \max_{b \in \mathcal{B}} G_b(\mathcal{N})$ slots, where*

$$G_b(\mathcal{N}) := \sum_{k=1}^{K} \frac{N_{\ell_b}^{(k)} w_k}{R_{\ell_b}} + \sum_{k=1}^{K} \frac{N_b^{(k)} w_k}{R_k^a} + \sum_{b': p(b')=b} \sum_{k=1}^{K} \frac{N_{\ell_{b'}}^{(k)} w_k}{R_{\ell_{b'}}} \tag{20}$$

2) *Each flow of class k receives a flow-rate of at least $w_k \left(\max_{b \in \mathcal{B}} G_b(\mathcal{N})\right)^{-1}$ bits/slot.*

A. Rate guarantee via distributed admission control

Consider a scenario where q_k bits/slot is the flow-rate requirement for a class $k \in [1:K]$, *i.e.* inelastic traffic. This rate requirement can be guaranteed under the slot reservation framework by incorporating admission control as follows.

Suppose the current network state is $\mathcal{N}$, and the rate requirement is satisfied for all the existing flows. Consider a new UE request of class $k \in [1:K]$ arriving at gNB b. Let $\{b_i\}_{i=1}^{I}$ denote the set of gNBs in the path from b to r, such that $b_1 = p(b)$, $b_{i+1} = p(b_i)$ for $i \in [1: I-1]$. Let $b_0 := b$,

This article has been accepted for publication in IEEE/ACM Transactions on Networking. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TNET.2023.3273349

and also note that $b_I = r$. We propose that the new UE is admitted into the network if and only if (21)-(22) hold.

$$G_b(\mathcal{N}) + \frac{w_k}{R_k^a} + \frac{w_k}{R_{\ell_b}} \leq \min_{k' \in [1,K]} \frac{w_{k'}}{q_{k'}} \tag{21}$$

$$G_{b_i}(\mathcal{N}) + \frac{w_k}{R_{\ell_{b_i}}} + \frac{w_k}{R_{\ell_{b_{i-1}}}} \leq \min_{k' \in [1,K]} \frac{w_{k'}}{q_{k'}} \ \forall i \in [1, I-1]$$

$$G_{b_I}(\mathcal{N}) + \frac{w_k}{R_{\ell_{b_{I-1}}}} \leq \min_{k' \in [1,K]} \frac{w_{k'}}{q_{k'}} \tag{22}$$

Suppose the new UE request is admitted. It will add to number of flows N_b^k at gNB b and to the number of flows on backhaul links ℓ_{b_i}, $N_{\ell_{b_i}}^{(k)}$, for $i \in [1 : I-1]$. Hence, this admission will change the value of $G_{b_i}(\cdot)$ at each gNB b_i for $i \in [0 : I]$. The LHS of (21)-(22) are the values of $\{G_{b_i}(.)\}_{i=0}^I$ if the UE is admitted. Since $G_{b'}(\cdot)$ is not affected for the rest of the gNBs $b' \in \mathcal{B} - \{b_i\}_{i=0}^I$, the rate guarantee follows from the second result (*i.e.* 2)) in Theorem 7.

For admission, the new UE request requires that the conditions (21)-(22) hold at gNBs in the path from r to b. Note that $G_{b_i}(\mathcal{N})$ can be evaluated locally at the gNB b_i, and a bit $x_i \in \{0, 1\}$ can be set to 1 if the condition holds at the gNB and 0 otherwise. For deciding admission, a 1 bit message x can be passed from b to r. At each gNB b_i, logical AND operation $\wedge$, is used to send message $x \wedge x_i$ to parent b_{i+1}, resulting in a 0 being delivered at the root gNB r if any of the conditions do not hold. Recall that in 3GPP IAB, the UE initial access procedure involves establishing such a forwarding route from r to b, and the message passing can be included in this procedure.

We also note that it is sensible to match the weights to requirements, *i.e.* choose $w_k = q_k$ for each $k = 1, \ldots, K$, although it is not necessary.

B. Latency guarantee via distributed admission control

The key idea in the previous Section IX-A is to achieve rate guarantees by limiting the window-size via admission control. This is possible since the flow rates are a function of the converged window size. We take the same approach to provide end-to-end packet latency guarantees in the following.

For each UE request at gNB b, in each window, there is a transfer of exactly τ_s bits on each hop in the path from r to the UE. Hence, the end-to-end latency is $I + 1$ times the window size, *i.e.* $(I+1) \max_{b \in \mathcal{B}} G_b(\mathcal{N})$. It is therefore clear that the admission control in the previous section can also be used to provide latency guarantees.

C. Guarantees under changing access link rates

In previous sections, we have mainly focused on a setup where the access rate of a UE request is fixed, from its arrival until its departure. However, on a long enough time scale, the mmWave link rates change due to fading and blocking[7]. The distributed admission control setup can be adapted for varying link rates, by allowing service interruptions, *i.e.* by dropping UE requests with bad link rates.

[7]See also the time varying model considered in Section VII

As earlier, suppose that the current network state is $\mathcal{N}$, and the minimum rate requirement is satisfied for all the existing UE requests. Here, we consider that all flows have the same weight 1 as in the DSR algorithm. Hence, a class k is only associated with a rate R_k^a, and a change of class is used to represent a change in access link rate.

1) Change in UE rate: Suppose an existing flow u at gNB b, changes from class k to class j, where $R_j^a > R_k^a$, *i.e.* improved rate. In this case, the UE request stays in the network, but the $\tau_{(b,u)}$ value is calculated with new access rate R_j^a, as τ_s / R_j^a.

Suppose an existing UE request at gNB b, changes from a class k to class j, where $R_j^a < R_k^a$. In this case, the UE request is only retained if (23) holds. Otherwise, the UE request is dropped by gNB b.

$$G_b(\mathcal{N}) - \frac{1}{R_k^a} + \frac{1}{R_j^a} \leq \frac{1}{q_j} \tag{23}$$

We note that in this case only $G_b(\cdot)$ is affected, and not $G_{b_i}(\cdot)$ for $i \in [1 : I]$, in contrast to the previous case. This is because the number of flows through the backhaul links in the path from r to b remains unchanged.

2) UE arrival with a good rate: Suppose a new UE request of class $k \in [1 : K]$ arrives at gNB b. The UE request is admitted if (21)-(22) holds as explained in section IX-A.

Suppose (21)-(22) does not hold. In this case, the new UE request can be still admitted by dropping an existing UE request with a smaller rate than R_k, and QoS is still maintained by (23). We propose that an existing UE request with smallest rate is dropped by gNB b.

X. Conclusions and Future Work

In the paper, we have provided a new distributed slot reservation framework for joint flow control and scheduling in mmWave IAB networks. We have shown it can be applied 1) in a static setup using our SSR algorithm, where it converges fast to the optimal solution, 2) in a dynamic setup with flow arrivals, using our DSR algorithm, which has a uniform delay performance across flows and also stability, and 3) for admission control to provide QoS guarantees in a dynamic setup with flow arrivals and changing link rates.

For the SSR algorithm, we have restricted to the tree topology of IAB networks and used its structure to derive the optimality result, Theorem 3. The other convergence result, Theorem 2, does not rely on the tree topology. Hence, in any network, the SSR algorithm always converges to a fixed point, even when the topology is not a tree, but the fixed point is not an optimal solution of LP (3) for all topologies. The question of clearing-time optimality for other topologies is left open. For future work it may also be important to consider gNBs that are capable of creating multiple beams.

In the dynamic model for DSR, we have assumed that long lasting blocking of access links does not happen on the time-scale of slot reservation and that the topology is fixed. Future work can consider topology reconfiguration due to blocking, in which blocking of access links causes UE requests to migrate to different gNBs. In such scenarios, the arrival and departure processes at different gNBs will be correlated, which

This article has been accepted for publication in IEEE/ACM Transactions on Networking. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TNET.2023.3273349

can affect the overall capacity region. It is also important in future work to consider channel models in which the data rates are time varying, due to blocking, and other channel impairments. We have provided illustrative results for DSR for such a model (Scenario 2) in Section VII, but designing an optimal slot reservation algorithm for dynamic rates remains an open problem.

Appendix A: Proofs of Theorem 1 and Theorem 2

Proof of Theorem 1. Suppose that $f^* > 1$. Let $\mathcal{F}$ denote the set of all the feasible vectors $\{f_S\}_{S\in\mathcal{S}} \in \mathbb{R}_+^{|\mathcal{S}|}$ such that $\sum_{S\in\mathcal{S}} f_S \leq 1$. Since $f^* > 1$, it follows that for any $\{f_S\} \in \mathcal{F}$ there exists a link $l \in \mathcal{L}$ such that $\gamma_l - \sum_{S:l\in S} f_S > 0$. Let

$$\delta := \inf_{\{f_S\}\in\mathcal{F}} \max_{l\in\mathcal{L}} (\gamma_l - \sum_{S:l\in S} f_S) \tag{24}$$

Consider any $T \geq 1$. Let F_S^T be the total number of slots allocated to a feasible set $S \in \mathcal{S}$, until time T, under an allocation policy. Define the time fraction allocated to feasible set S by $f_S^T := F_S^T/T$ and by feasibility we have that $\sum_{S\in\mathcal{S}} f_S^T \leq 1$, hence $\{f_S^T\}_{S\in\mathcal{S}} \in \mathcal{F}$.

Note that $\frac{1}{T}\sum_{t=0}^{T} U_l(t) = \sum_{S:l\in\mathcal{S}} f_S^T$. Hence from (24), there exists $l \in \mathcal{L}$ such that $\frac{1}{T}\sum_{t=0}^{T} U_l(t) \leq \gamma_l - \delta$ for each $T \geq 1$. Hence, $\boldsymbol{\alpha}$ is not achievable if $f^* > 1$. □

Proof of Theorem 2. We show that for each $l \in \mathcal{L}$, there are at least τ_l allocated slots in the interval $[t - w(t) + 2 : t + 1]$, which is enough to prove the theorem.

We categorize the links in $\mathcal{L}$ into mutually exclusive (and exhaustive) sets $\mathcal{L}_1, \mathcal{L}_2, \mathcal{L}_{3,1}, \mathcal{L}_{3,2}$. $\mathcal{L}_1$ is the set of links l such that there are at least $\tau_l + 1$ allocated slots in the interval $[t - w(t) + 1 : t]$. $\mathcal{L}_2$ is the set of links l for which there are exactly τ_l slots allocated to l in the interval $[t - w(t) + 1 : t]$ and for which $t - w(t) + 1$ is not an allocated slot of l. $\mathcal{L}_{3,1}$ is the set of links l such that there are exactly τ_l slots allocated to l in the interval $[t - w(t) + 1 : t]$, and both slot $t - w(t) + 1$ and slot t are slots allocated to link l. $\mathcal{L}_{3,2}$ is the set of links l such that there are exactly τ_l slots allocated to l in the interval $[t - w(t) + 1 : t]$, and slot $t - w(t) + 1$ is allocated to l but slot t is not allocated to link l.

It is clear that at least τ_l slots are allocated to link l in the interval $[t - w(t) + 2 : t + 1]$ for any link $l \in \mathcal{L}_1$. For a link $l \in \mathcal{L}_2$, there are exactly τ_l allocated slots in the interval $[t - w(t) + 2 : t]$, and hence at least τ_l allocated slots in the interval $[t - w(t) + 2 : t + 1]$ also. Now we consider the two other sets of links.

Case- $l \in \mathcal{L}_{3,1}$: For a link $l \in \mathcal{L}_{3,1}$, we consider two subcases: 1) t is an ending slot for link l, 2) t is not an ending slot for link l.

Subcase 1): In this subcase, t is an ending slot for l, so the interval of slots $[t - \tau_l + 1 : t]$ is allocated to l. Since l is in $\mathcal{L}_{3,1}$, slot $t - w(t) + 1$ is allocated to l. Since exactly τ_l slots are allocated to l in $[t - w(t) + 1 : t]$ it follows that $w(t) = \tau_l$. But any other link $k \in \mathcal{L} - \{l\}$ has all its required τ_k slots allocated in $[t - w(t) + 1 : t]$ so it follows that in this subcase, $l \notin I(k), \forall k \in \mathcal{L} - \{l\}$ (*i.e.* link l does not interfere with any other link). Hence, after ending slot t, link l will book slots $[t + 1 : t + \tau_l]$. Thus, there are τ_l allocated slots in the interval $[t - w(t) + 2 : t + 1]$ in this subcase.

Subcase 2): Since slots are allocated as a contiguous block, and since slot t, occupied by l, is not an ending slot for l (in this subcase), it follows that $t + 1$ is also allocated to l. Therefore, there are τ_l allocated slots in the interval $[t - w(t) + 2 : t + 1]$ in this subcase.

Case- $l \in \mathcal{L}_{3,2}$: Since there are exactly τ_l allocated slots in the interval $[t - w(t) + 1 : t]$, it follows that there is one ending slot for l in the interval. Let t' denote the ending slot for l in the interval $[t - w(t) + 1 : t]$. We note that $t' \neq t$ since by definition of $\mathcal{L}_{3,2}$, slot t is not allocated to link l.

Suppose first that all of the allocated slots to link l from the update at slot t' lie inside the interval $[t - w(t) + 1 : t]$. This is impossible, for if so, there are the τ_l slots from the update at slot t', plus slot t' itself, allocated to l, all in the interval $[t - w(t) + 1 : t]$. This contradicts the fact that there are exactly τ_l slots allocated to the link in the interval $[t - w(t) + 1 : t]$.

Suppose next that some of the τ_l slots from the update at t' lie outside the interval $[t - w(t) + 1 : t]$, and others lie inside. Due to the contiguity of slot allocation (in contiguous blocks) this implies slot t is occupied by l, which contradicts the fact that l does not occupy slot t (from the definition of $\mathcal{L}_{3,2}$).

We conclude that all the allocated slots to link l from the update at t' lie strictly outside the interval $[t - w(t) + 1 : t]$. Thus, the slots allocated to link l in the interval $[t - w(t) + 1 : t]$ all lie in the sub-interval $[t - w(t) + 1 : t - w(t) + \tau_l]$ (which includes $t' = t - w(t) + \tau_l$) in this case.

Note that for each $k \in I(l)$, there are at least τ_k allocated slots in the interval $[t - w(t) + 1 : t]$. Since the slots $[t - w(t) + 1 : t - w(t) + \tau_l]$ are occupied by l, there are at least τ_k allocated slots in the interval $[t - w(t) + \tau_l + 1 : t]$ for each $k \in I(l)$. It follows that the occupied slots (by $k \in I(l)$) during the update for l at time $t' = t - w(t) + \tau_l$ must satisfy $\bigcup_{k\in I(l)} T_k(t') = [t - w(t) + \tau_l + 1 : t]$.

It follows that the immediate free slot for link l during update at ending slot t' must be $t + 1$. Hence, during this update, l occupies the slots $[t + 1 : t + \tau_l - 1]$. Hence, it follows that for each $l \in \mathcal{L}_{3,2}$, there are exactly τ_l slots in the interval $[t - w(t) + 2 : t + 1]$. This concludes the proof. □

Appendix B: Proof of Theorem 3

In this section, we describe several key properties of the SSR algorithm, and provide the proof of Theorem 3. First, we introduce the necessary notation and prove preliminary results which are required.

Definition 5. *A* clique set C_b*, for a gNB* b*, is the set of links connected to* b *in* G*,* i.e. $C_b := \{l_b\} \bigcup \mathcal{L}_b$.

Lemma 1 is an important step in the proof of Theorem 3. It concerns the optimal value of the linear program in (3).

Lemma 1. *The optimal value of LP (3),* $f^* \geq \max_{b\in\mathcal{B}} \sum_{l\in C_b} \gamma_l$.

Proof. Consider a $b \in \mathcal{B}$. Note that each $l \in C_b$ conflicts with other links in $C_b - \{l\}$. Hence, each feasible set $S \in \mathcal{S}$ can

contain at most one link in C_b. Let $\mathcal{S}_l$ denote the feasible sets which contain a link $l \in C_b$. We obtain that

$$\sum_{S\in\mathcal{S}} f_S^* \geq \sum_{l\in C_b}\sum_{S\in\mathcal{S}_l} f_S^* \tag{25}$$

From the constraints of LP (3), it is clear that $\sum_{S\in\mathcal{S}_l} f_S^* \geq \gamma_l$ for each $l \in C_b$. The Lemma is now immediate from (25). □

We now introduce the necessary definitions and the preliminary results concerning the SSR algorithm required in the proof of Theorem 3. We define $\epsilon_l(t)$ to be the earliest ending slot of l which is greater than or equal to t as follows.

$$\epsilon_l(t) := \min_{n\in\mathbb{N}}\{s_l^n + \tau_l - 1 : s_l^n + \tau_l - 1 \geq t\} \tag{26}$$

For $t > s_l^1 + \tau_l - 1$, we define $\beta_l(t)$, the latest ending slot of l which is less than t as follows.

$$\beta_l(t) := \max\{k < t : \epsilon_l(k) = k\} \tag{27}$$

We define the *inter-update gap* of a link $l \in \mathcal{L}$ as

$$g_l(t) := \epsilon_l(t) - \beta_l(t) \tag{28}$$

for $t > s_l^1 + \tau_l - 1$. We present the following lemma which provides a relation between the window size $w(t)$ (defined in (6)) and the inter-update gaps.

Lemma 2. *At any* $t \geq \max_{l\in\mathcal{L}} s_l^1 + \tau_l - 1$,

$$w(t) \leq \max_{l\in\mathcal{L}} g_l(t) \tag{29}$$

Proof. Consider a $l \in \mathcal{L}$. Let $n \in \mathbb{N}$ be such that t lies in the interval $[s_l^n + \tau_l : s_l^{n+1} + \tau_l - 1]$. Then $\beta_l(t) = s_l^n + \tau_l - 1$, $\epsilon_l(t) = s_l^{n+1} + \tau_l - 1$ and $g_l(t) = s_l^{n+1} - s_l^n$.

If $t < s_l^{n+1}$ then $t - g_l(t) + 1 = t - s_l^{n+1} + s_l^n + 1 \leq s_l^n$. So in this case $[s_l^n : s_l^n + \tau_l - 1]$ is entirely contained in the interval $[t - g_l(t) + 1 : t]$, and $[s_l^{n+1} : s_l^{n+1} + \tau_l - 1]$ is entirely outside the same interval. It follows that link l has exactly τ_l allocated slots in the interval $[t - g_l(t) + 1 : t]$.

If $s_l^{n+1} \leq t \leq s_l^{n+1} + \tau_l - 1$ then

$$s_l^n + 1 \leq t - g_l(t) + 1 \leq s_l^n + \tau_l.$$

In this case, link l is allocated all the slots between slot $t - g_l(t) + 1$ and slot $s_l^n + \tau_l - 1$. This is a total of $\tau_l - 1 + s_l^{n+1} - t$ slots allocated to l in this interval. In this case link l is also allocated the $t - s_l^{n+1} + 1$ slots between s_l^{n+1} and t. Adding the two parts together, there is a total of exactly τ_l slots allocated to l in the interval $[t - g_l(t) + 1 : t]$.

It follows that each link l has at least τ_l allocated slots in the interval $[t - \max_{l'\in\mathcal{L}} g_{l'}(t) + 1 : t]$. The Lemma is immediate from the definition of window size $w(t)$. □

Definition 6. *A link* l *is said to have a* large gap *at time* t*, if*

$$g_l(t) > \max_{b\in\mathcal{B}} \sum_{l\in C_b} \tau_l =: \tau_{\mathcal{B}}^{\max} \tag{30}$$

The following Lemma 3 regarding the ending slots of neighboring links is an intermediate lemma required in the proof of the following Lemma 4.

Lemma 3. *Suppose that* t *is an ending slot of a link* $l = (n_0, m_0)$ *such that* $t \geq s_l^1 + \tau_l$*. Then exactly one of the following conditions must hold*

1) $t - \tau_l$ *is an ending slot for link* l*.*
2) $t - \tau_l$ *is an ending slot for a link* $k \in I(l)$*. Further,* k *is an element of exactly one of the sets* $C_{n_0} - \{l\}$*,* $C_{m_0} - \{l\}$*.*

Proof of Lemma 3. Let the ending slot t be $s_l^{n+1} + \tau_l - 1$, *i.e.* ending slot belonging to $n+1$-th allocation block of l. Let $t' = \beta_l(t) = s_l^n + \tau_l - 1$ be the previous ending slot. Note that $s_l^{n+1} \geq s_l^n + \tau_l$, since slots $[s_l^n : s_l^n + \tau_l - 1]$ is the n-th allocation block of l. We consider the following two cases. **Case 1** - $s_l^{n+1} = s_l^n + \tau_l$: i.e, slot $t' + 1$ is free during the update at t'. In this case, $t' = t - \tau_l$ is an ending slot for link l. Statement 1) of Lemma 3 holds in this case.

Case 2 - $s_l^{n+1} > s_l^n + \tau_l$: In this case, during the update at t', $s_l^{n+1} - 1$ is a slot occupied by a link (say k) in $I(l)$ (see line 1 of Algorithm 1). Note that link k occupies slot $t - \tau_l = s_l^{n+1} - 1$, but not $t - \tau_l + 1 = s_l^{n+1}$ (since it is occupied by link l). Hence, $t - \tau_l$ is an ending slot of link k. Finally, note that $I(l) = C_{n_0} - \{l\} \bigcup C_{m_0} - \{l\}$ and $C_{n_0} - \{l\} \bigcap C_{m_0} - \{l\} = \phi$. Statement 2) of Lemma 3 holds in this case. □

The following Lemma 4 is the key basis for inductive step in the proof of Lemma 5 below.

Lemma 4. *Suppose* t *is an ending slot of a link* $l \in \mathcal{L}$ *such that* $g_l(t) > \tau_{\mathcal{B}}^{\max}$ *and, for which* $t \geq s_l^1 + \tau_l$ *and* $t - \tau_l \geq \max_{l'\in I(l)} s_{l'}^1 + \tau_{l'}$*. Then there exists* $k \in I(l)$ *such that*
a) $t - \tau_l$ *is an ending slot of* k *(and hence not of link* l*).*
b) $g_k(t - \tau_l) > \tau_{\mathcal{B}}^{\max}$*.*
c) $t - \tau_l - \tau_k$ *is not an ending slot of* l*.*
d) $\exists j \in I(k) - \{l\} : t - \tau_l - \tau_k$ *is an ending slot of* j*.*

Proof of Lemma 4. It follows from Lemma 3 that $t - \tau_l$ is an ending slot for either link l or for some link $k \in I(l)$. If $t - \tau_l$ is an ending slot for link l, then $g_l(t) = t - (t - \tau_l) = \tau_l \leq \tau_{\mathcal{B}}^{\max}$, which is a contradiction (since it is given that $g_l(t) > \tau_{\mathcal{B}}^{\max}$). Hence, $t - \tau_l$ is an ending slot for some link $k \in I(l)$, *i.e.* a).

Let $t' = \beta_l(t)$, *i.e.* t' is the latest ending slot of l before t. We have established that during the update at t', $t - \tau_l$ is already allocated to some link $k \in I(l)$. Let $I^* \subseteq I(l)$ denote the set of all the links in $I(l)$ that are occupying the slot $t - \tau_l$. In the following, we will show the existence of a link $k \in I^*$ that satisfies properties $b), c)$ & $d)$ of Lemma 4. We note that $a)$ follows trivially since $t - \tau_l$ is an ending slot for all links $l' \in I^*$. The existence of such a link $k \in I^*$ follows from Claim A and Claim B below.

Claim (Claim A)**.** *For each link* $k \in I^*$*,* $t - \tau_l - \tau_k$ *is not an ending slot of link* l*, but is an ending slot of a link* $j \in \{k\} \bigcup I(k) - \{l\}$*.*

Proof of Claim A. For each $k \in I^*$, applying Lemma 3 at time $t - \tau_l$, we have that $t - \tau_l - \tau_k$ is an ending slot for a link in the set $\{k\} \bigcup I(k)$. Note that $l \in I(k)$ for each $k \in I^*$. Now assume that there exists a link $k \in I^*$, such that $t - \tau_l - \tau_k$ is an ending slot of link l. It then follows that

$$g_l(t) \leq t - (t - \tau_l - \tau_k) = \tau_l + \tau_k \leq \tau_{\mathcal{B}}^{\max}$$

which is a contradiction since it is given that $g_l(t) > \tau_{\mathcal{B}}^{\max}$. Hence, $t - \tau_l - \tau_k$ is not an ending slot of l for any $k \in I^*$. This completes the proof of Claim A.

We now obtain the following Claim B, which will then be used to show that there exists a link $k \in I^*$ satisfying properties b) and d) of Lemma 4. c) is immediate from the above Claim A.

Claim (Claim B)**.** *There exists a link $k \in I^*$ such that there is no ending slot of k in $[t' + 1 : t - \tau_l - \tau_k]$.*

Proof of Claim B: Suppose not. Assume all links $k \in I^*$ have at least one ending slot in the interval $[t' + 1 : t - \tau_l - \tau_k]$. Hence, each link $k \in I^*$ is allocated the slot $t - \tau_l$ during an update in interval $[t' + 1 : t - \tau_l - \tau_k]$, and not before. Therefore, during link l's update at time t', $t - \tau_l \notin T_k(t')$ for any $k \in I(l)$, *i.e.* $t - \tau_l$ has not been allocated to any link in $I(l)$ at time t'. No slots later than $t - \tau_l$ have been allocated either. Thus, a block of slots including slot $t - \tau_l$ would be allocated to link l at time t'. This is a contradiction, since it is already established that $t - \tau_l$ is allocated to a link $k \in I(l)$ (in the first paragraph of the proof of this lemma). This completes the proof of Claim B.

Since $[t' - \tau_l + 1 : t']$ are occupied by l, the latest ending slot of the link k (the one in the statement of Claim B), before the ending slot $t - \tau_l$, is $\beta_k(t - \tau_l) \leq t' - \tau_l$. Therefore,

$$\begin{aligned} g_k(t - \tau_l) &= t - \tau_l - \beta_k(t - \tau_l) \\ &\geq t - t' = g_l(t) > \tau_{\mathcal{B}}^{\max}. \end{aligned}$$

Hence, we obtain statement b) of Lemma 4. It also follows from Claim B that slot $t - \tau_l - \tau_k$ is not an ending slot of link k. Statement d) now follows from Claim A, since $k \in I^*$. □

The following Lemma 5 shows that if large gaps exist for two links connected to a node n_0 at certain specified times, then there exists two links connected to a node n_1, a neighbour of node n_0, which also have large gaps at earlier times than the first two links.

Lemma 5. *Suppose there exist links $i_0 = (n_{-1}, n_0)$ and $j_0 = (n_0, n_0')$ sharing a common node n_0, and $t_0 > \sum_{\ell \in C_{n_0}} \tau_\ell + \max_{l \in \mathcal{L}} s_l^1 + \tau_l - 1$, such that a) and b) hold.*
a) t_0 is an ending slot of i_0 and $t_1 = t_0 - \tau_{i_0}$ is an ending slot of j_0.
b) $g_{i_0}(t_0) > \tau_{\mathcal{B}}^{\max}$ and $g_{j_0}(t_1) > \tau_{\mathcal{B}}^{\max}$.
Then, there exist links $i_1 = (n_0, n_1) \in C_{n_0} - \{i_0\}$ and $j_1 = (n_1, n_1') \in C_{n_1} - \{i_1\}$, and a time t' satisfying $t_0 - \sum_{\ell \in C_{n_0}} \tau_\ell \leq t' < t_0$ such that 1) and 2) hold.
1) t' (and $t' + \tau_{i_1}$) is an ending slot of j_1 (and i_1 respectively).
2) $g_{j_1}(t') > \tau_{\mathcal{B}}^{\max}$ and $g_{i_1}(t' + \tau_{i_1}) > \tau_{\mathcal{B}}^{\max}$.
(see Figure 8(a) for an illustration).

Proof. Before starting the proof, we make some preliminary remarks. The proof is inductive, and will generate a set of links $\{l_a\}_{a=0}^{N-1}$, all connected to node n_0 (hence in the set C_{n_0}), with $N \leq |C_{n_0}|$. The first two elements in this set are given by $l_0 = i_0$ and $l_1 = j_0$.

For a link $l_a := (n_0, m) \in C_{n_0}$ and time t', we say that $S_1(l_a, t')$ holds if $\exists \ell \in C_m - \{l_a\} : g_\ell(t') > \tau_{\mathcal{B}}^{\max}$.

For a set of links $A \subseteq C_{n_0}$ and time t', we say that $S_2(A, t')$ holds if $\exists \ell \in C_{n_0} - A : g_\ell(t') > \tau_{\mathcal{B}}^{\max}$.

The proof below contains steps 1 to $N - 1$. At each step a, we will show that there are two possibilities either $S_1(l_a, t_a - \tau_{l_a})$ holds, or $S_2(A_a, t_a - \tau_{l_a})$ holds, where $A_a := \{l_p\}_{p=0}^a$. If $S_1(l_a, t_a - \tau_{l_a})$ holds, we will show that the proof is completed. Otherwise, if $S_2(A_a, t_a - \tau_{l_a})$ holds, it implies a further step $a+1$ is needed, where the next link, l_{a+1}, is defined. Again, we show that either $S_1(l_{a+1}, t_{a+1} - \tau_{l_{a+1}})$ holds, or $S_2(A_{a+1}, t_{a+1} - \tau_{l_{a+1}})$. The proof is completed if $S_1(l_a, t_a - \tau_{l_a})$ holds at any step a, and proceeds to step $a+1$ otherwise. To complete the proof, we show that the process has to terminate at a step $N \leq |C_{n_0}|$. Rather than enumerate all steps explicitly, we use mathematical induction.

The proof is presented as follows.

Step 1: We start with $l_0 := i_0$, $l_1 := j_0$, $m_1 = n_0'$, $t_0 := t$. and $t_1 = t_0 - \tau_{l_0}$. It follows from Lemma 4 (identifying link l in Lemma 4 with link l_1 here) that there exists a link $k \in I(l_1)$ such that $t_1 - \tau_{l_1}$ is an ending slot of link k, but not of link l_1, and $g_k(t_1 - \tau_{l_1}) > \tau_{\mathcal{B}}^{\max}$. Either $k \in C_{m_1}$ or $k \in C_{n_0}$. In the former case, $k = (m_1, m_1')$ for some node m_1' connected to m_1 in the tree, and we have $S_1(l_1, t_1 - \tau_{l_1})$ holding. In this case, the result of the lemma holds, with $i_1 = l_1 = j_0$, $n_1 = m_1$, $j_1 = k$, $n_1' = m_1'$. In the case $k \in C_{n_0}$, we have $k \neq l_0$ and $k \neq l_1$ since l_0 and l_1 have no ending slots in the interval $\left[t_0 - \tau_{C_{n_0}} : t_1\right]$. But $k \in C_{n_0}$ so we can define m_2 by $k = (n_0, m_2)$. Defining $A_1 := \{l_0, l_1\}$, we have that $S_2(A_1, t_1 - \tau_{l_1})$ holds. In this case, to proceed to the next step, we define $l_2 := k$, $A_2 := A_1 \bigcup \{l_2\}$, $t_2 := t_1 - \tau_{l_1} = t_0 - \sum_{p=0}^1 \tau_{l_p}$. Note that l_2 does not have an ending slot in the interval $\left[t_0 - \tau_{C_{n_0}} : t_2\right]$, since $g_k(t_2) > \tau_{\mathcal{B}}^{\max}$, and neither do the other links in A_2, as observed earlier.

Now we state the inductive hypothesis. Suppose the links in $A_a = \{l_0, l_1, \ldots l_a\}$ have been generated, with ending slots given by $t_p = t_{p-1} - \tau_{l_{p-1}} = t_0 - \sum_{p'=0}^{p-1} \tau_{l_{p'}}$, $p = 1, 2, \ldots a$ (with t_0 given as in statement of the lemma), none of the links $l_0, l_1, \ldots l_a$ have an ending slot in the interval $\left[t_0 - \tau_{C_{n_0}} : t_a - \tau_{l_a}\right]$, none of $S_1(l_p, t_{p+1})$ are true for $p = 1, 2, \ldots, a-1$, but each of $S_2(A_p, t_{p+1})$ are true for $p = 1, 2, \ldots, a-1$. Each link l_p is connected to node n_0, and m_p have been defined such that $l_p = (n_0, m_p)$, $p = 1, 2, \ldots a$.

It follows from Lemma 4 (identifying link l in the Lemma with link l_a here) that there exists a link $k \in I(l_a)$ such that t_{a+1} is an ending slot of link k, but not of link l_a, and $g_k(t_{a+1}) > \tau_{\mathcal{B}}^{\max}$, where we define $t_{a+1} = t_a - \tau_{l_a} = t_0 - \sum_{p=0}^a \tau_{l_p}$. Either $k \in C_{m_a}$ or $k \in C_{n_0}$. In the former case, $k = (m_a, m_a')$ for some node m_a' connected to m_a in the tree, and we have $S_1(l_a, t_{a+1})$ holding. In this case, the result of the lemma holds, with $i_1 = l_a$, $n_1 = m_a$, $j_1 = k$, and $n_1' = m_a'$. In the case $k \in C_{n_0}$, we have $k \notin \{\ell_0, \ell_1, \cdots \ell_a\}$ since none of $l_0, l_1, \ldots l_a$ have an ending slot in the interval $\left[t_0 - \tau_{C_{n_0}} : t_{a+1}\right]$. This means that $S_2(A_{a+1}, t_{a+1})$ holds, where we define $l_{a+1} := k$, and $A_{a+1} = A_a \bigcup \{l_{a+1}\}$.

The result of the lemma follows by induction, because the above steps must terminate at the latest when $a = |C_{n_0}| - 1$, for then all the links l_p connected to n_0 will have been exhausted, but Lemma 4 still applies, so in this case $S_1(l_a, t_{a+1})$ must be true for $t_{a+1} = t_0 - \sum_{\ell \in C_{n_0}} \tau_\ell$.

□

Lemma 5 states that, under the conditions of the Lemma, the links $i_1 = (n_0, n_1)$ and $j_1 = (n_1, n_1')$ must exist, satisfying the properties stated in the Lemma. This means that the conditions of the Lemma are not compatible with node n_0 being the parent node of only leaf nodes in graph G, with node n_{-1} being the parent of node n_0 in graph G. This observation will be important in the Proof of Theorem 3 below.

A. Proof of Theorem 3

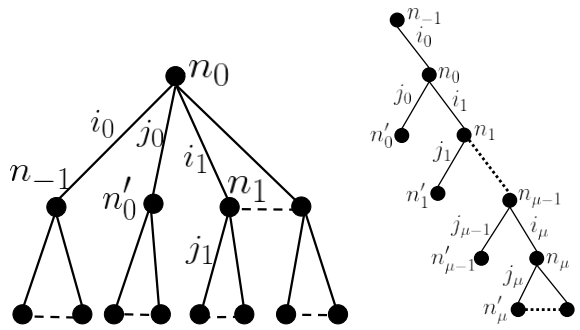
Proof of Theorem 3. For the proof, we will show that $\exists T \leq T'$ such that $g_l(T) \leq \tau_{\mathcal{B}}^{\max}$ for each $l \in \mathcal{L}$. It follows from Lemma 2 that $w(T) \leq \tau_{\mathcal{B}}^{\max}$.

Since no two links in a clique set C_n, for any node n, can be allocated in the same slot, it follows that at least $\sum_{l \in C_n} \tau_l$ slots are needed for links in C_n. Hence, at least $\tau_{\mathcal{B}}^{\max}$ are needed to allocate τ_l slots for each link $l \in \mathcal{L}$. Since $w(T) \leq \tau_{\mathcal{B}}^{\max}$, the result follows.

We therefore need to prove the statement $\exists T \leq T'$ such that $g_l(T) \leq \tau_{\mathcal{B}}^{\max}$ for each $l \in \mathcal{L}$, which we do by contradiction. We suppose that there exists a link i_0 with an ending slot t_0 such that $g_{i_0}(t_0) > \tau_{\mathcal{B}}^{\max}$ for some $t_0 > \sum_{l \in \mathcal{L}} \tau_l$. We will show below that this supposition leads to a contradiction. For the argument below, we first assume that this link, with the stated property, exists.

Using Lemma 4, in which we identify the link l in Lemma 4 with i_0 and the time t in the lemma with t_0 here, there exists a link $k \in I(i_0)$ such that $g_k(t_0 - \tau_{i_0}) > \tau_{\mathcal{B}}^{\max}$, and $t_0 - \tau_{i_0}$ is the ending slot of link k. Let n_0 denote the shared node between links i_0 and k. Now we apply Lemma 5 to this case. To do this, we denote link k by j_0 to be consistent with the notation of Lemma 5. Thus, we have $i_0 = (n_{-1}, n_0)$ for some node n_{-1}, $j_0 = (n_0, n_0')$, and the conditions a) and b) of Lemma 5 hold. Thus, by Lemma 5 there exists links i_1 and j_1 satisfying conditions 1) and 2) of the lemma. Treating n_0 as the root node of graph G, we have the situation depicted in Figure 8(a). It can be that $j_0 = i_1$ and $n_0' = n_1$, but not necessarily.

Since conditions a) and b) of Lemma 5 apply to links i_1 and j_1, we can apply Lemma 5 again. This process can be repeated until there are no more links in G available to obtain new i_1 and j_1 links. Indeed, after μ such steps (with $\mu \leq |\mathcal{L}|$) we have selected and labeled certain links in the tree G as depicted in Figure 8(b) (the depicted tree is of course a subgraph of G). Eventually, at some point on or before μ reaching the value $|\mathcal{L}|$, we must have the situation holding that n_μ is the parent of only leaf nodes in the tree. This must happen because G is finite. In this case, n_μ' is a child node of parent node n_μ and hence n_μ' is a leaf node. However, the conditions of Lemma 5 hold, taking i_μ here to represent i_0 and j_μ here to represent j_0, for i_0 and j_0 as stated in Lemma 5. Lemma 5 then guarantees the existence of links $i_{\mu+1}$ and $j_{\mu+1}$ satisfying the conditions a) and b) satisfied by the i_1 and j_1 in Lemma 5. But this is a contradiction, because n_μ is the parent of only leaf nodes in the tree. This contradiction implies that the original assumption that there exists a link i_0 with an ending slot t_0 such that $g_{i_0}(t_0) > \tau_{\mathcal{B}}^{\max}$ for some $t_0 > T$, cannot be true. The statement of the theorem then holds by the arguments given in the first two paragraphs of this proof. □



(a) At step 1 (b) At step μ

Fig. 8. Example diagram for proof of Theorem 3.

Appendix C: Proofs of Theorem 5 and Theorem 6

Proof of Theorem 5. Suppose (16) holds for $b \in B$. Let $\mathcal{U}_b^{(k)}(t_c)$ denote the set of active UE requests of class k at gNB $b \in \mathcal{B}$ during time t_c. Consider the following state evolution equations.

$$N_b^{(k)}(t_c) = N_b^{(k)}(0) + \mathcal{P}_{\nu_b^{(k)}}(t_c) - \mathcal{P}_{1/D}\left(\int_0^{t_c} \alpha^{(k)}(s)ds\right)$$

for $k = 1, \ldots, K$, where $\mathcal{P}_\lambda(.)$ represents the Poisson process with parameter λ, and $\alpha^{(k)}(t_c) = \sum_{u \in \mathcal{U}_b^{(k)}(t_c)} \alpha_u(t_c)$. Taking expectations, we obtain $\mathbb{E}[N_b^{(k)}(t_c)]$

$$= \mathbb{E}[N_b^{(k)}(0)] + \nu_b^{(k)} t_c - \frac{1}{D}\int_0^{t_c} \alpha^{(k)}(s)ds. \tag{31}$$

For each backhaul link $\ell_{b'}$, $b' \in \mathcal{B}$, let $\mathcal{U}_{\ell_{b'}}(t_c) := \{u \in \mathcal{U}(t_c) : \ell_{b'} \in P_u\}$ be the set of flows that are using link ℓ_b. Note that $N_{\ell_{b'}}(t_c) = |\mathcal{U}_{\ell_{b'}}(t_c)|$. Hence,

$$\mathbb{E}[N_{\ell_{b'}}(t_c)] = \mathbb{E}[N_{\ell_{b'}}(0)] + \nu_{\ell_{b'}} t_c - \frac{1}{D}\int_0^{t_c} \sum_{u \in \mathcal{U}_{\ell_{b'}}(s)} \alpha_u(s)ds$$

$$\frac{\mathbb{E}[N_{\ell_{b'}}(t_c) - N_{\ell_{b'}}(0)]}{R_{\ell_{b'}}} = \frac{\nu_{\ell_{b'}} t_c}{R_{\ell_{b'}}} - \frac{1}{D}\int_0^{t_c} \gamma_{\ell_{b'}}(s)ds \tag{32}$$

Note that $\mathcal{L}_b(t_c)$ consists of backhaul links $\{\ell_{b'}\}_{b' \in \mathcal{B}: p(b') = b}$ and access links $\bigcup_{k=1}^K \{(b,u)\}_{u \in \mathcal{U}_b^{(k)}(t_c)}$ of gNB b. Hence, from (31) and (32), we obtain $\mathbb{E}[F_b(\mathcal{N}(t_c))] - \mathbb{E}[F_b(\mathcal{N}(0))]$

$$= t_c \left(\sum_{k=1}^K \frac{\nu_b^{(k)}}{R_k^a} + \frac{\nu_{\ell_b}}{R_{\ell_b}} + \sum_{b': p(b') = b} \frac{\nu_{\ell_{b'}}}{R_{\ell_{b'}}}\right) - \frac{1}{D}\int_0^{t_c} \sum_{l \in \mathcal{L}_b^{(s)} \bigcup \{\ell_b\}} \gamma_l^{(s)} ds \tag{33}$$

The first term $> t_c/D$ from (16), and the second term $\leq t_c/D$ from (12). Hence, $\exists \delta_1 > 0$ such that $\mathbb{E}[F_b(\mathcal{N}(t_c))] - \mathbb{E}[F_b(\mathcal{N}(0))] \geq \delta_1 t_c/D$. Since this is true for each $t_c \geq 0$, the Markov process is not ergodic. □

This article has been accepted for publication in IEEE/ACM Transactions on Networking. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TNET.2023.3273349

Proof of Theorem 6. The proof consists of showing that the *fluid limit* of $\{\mathcal{N}(t_c)\}_{t_c\in\mathbb{R}_+}$ is stable. The fluid limit of the setup is characterized by the absolutely continuous trajectories $\{\bar{\mathcal{N}}(t_c)\}_{t_c\in\mathbb{R}_+} \in \mathcal{C}(R_+^{|\mathcal{B}|\times K})$, which satisfy the following differential equations

$$\frac{d\bar{N}_b^{(k)}(t_c)}{dt_c} = \nu_b^{(k)} - \frac{1}{D}\frac{\bar{N}_b^{(k)}(t_c)}{\bar{N}_{\mathcal{B}}^{\max}(t_c)} \tag{34}$$

for $k \in [1:K]$ and $b \in \mathcal{B}$, where $\bar{N}_{\mathcal{B}}^{\max}(t_c) = \max_{b\in\mathcal{B}} F_b(\bar{\mathcal{N}}(t_c))$, and with $\sum_{b\in\mathcal{B}}\sum_{k=1}^{K}\bar{N}_b^{(k)}(0) \le 1$ for the initial conditions on $\bar{\mathcal{N}}(0)$.

The fluid limit is stable provided $\exists T > 0$ such that $\bar{\mathcal{N}}(t_c) = \mathbf{0}, \forall t_c \ge T$ for all such initial conditions.

We now provide the key arguments for fluid limit stability. For $b \in \mathcal{B}$ and $\bar{\mathcal{N}} \in \mathbb{R}_+^{|\mathcal{B}|\times K}$, recall

$$F_b(\bar{\mathcal{N}}) := \sum_{k=1}^{K}\frac{\bar{N}_b^{(k)}}{R_k^a} + \frac{\bar{N}_{\ell_b}}{R_{\ell_b}} + \sum_{b':p(b')=b}\frac{\bar{N}_{\ell_{b'}}}{R_{\ell_{b'}}} \tag{35}$$

Hence from (34), $\frac{dF_b(\bar{\mathcal{N}}(t_c))}{dt_c}$ equals

$$\sum_{k=1}^{K}\frac{\nu_b^{(k)}}{R_k^a} + \frac{\nu_{\ell_b}}{R_{\ell_b}} + \sum_{b':p(b')=b}\frac{\nu_{\ell_{b'}}}{R_{\ell_{b'}}} - \frac{1}{D}\frac{F_b(\bar{\mathcal{N}}(t_c))}{\bar{N}_{\mathcal{B}}^{\max}(t_c)} \tag{36}$$

Let

$$\delta' := \min_{b\in\mathcal{B}}\frac{1}{D} - \left(\sum_{i=1}^{K}\frac{\nu_b^i}{R_i^a} + \sum_{b':p(b')=b}\frac{\nu_{l_{b'}}}{R_{l_{b'}}} + \frac{\nu_{l_b}}{R_{l_b}}\right) > 0$$

Since $\bar{N}_{\mathcal{B}}^{\max}(t_c) = \max_{b\in\mathcal{B}} F_b(\bar{\mathcal{N}}(t_c))$ and $\bar{N}_{\mathcal{B}}^{\max}(t_c)$ is absolutely continuous, the derivative $\frac{dF_b(\bar{\mathcal{N}}(t_c))}{dt_c} \le -\delta'$, whenever $\bar{N}_{\mathcal{B}}^{\max}(t_c) = F_b(\bar{\mathcal{N}}(t_c))$ for almost every t_c. Since $\bar{N}_{\mathcal{B}}^{\max}(t_c) = \max_{b\in\mathcal{B}} F_b(\bar{\mathcal{N}}(t_c))$, using similar arguments, we obtain

$$\frac{d\bar{N}_{\mathcal{B}}^{\max}(t_c)}{dt_c} \le -\delta' \tag{37}$$

for almost every t_c such that $\bar{N}_{\mathcal{B}}^{\max}(t_c) > 0$. Hence, the fluid limit stability can be obtained by using $\bar{N}_{\mathcal{B}}^{\max}(t_c)$ as a Lyapunov function. □

REFERENCES

[1] X. Lin and N. Shroff, "The impact of imperfect scheduling on cross-layer congestion control in wireless networks," *IEEE/ACM Transactions on Networking*, vol. 14, no. 2, pp. 302–315, 2006.

[2] X. Lin, N. Shroff, and R. Srikant, "A tutorial on cross-layer optimization in wireless networks," *IEEE Journal on Selected Areas in Communications*, vol. 24, no. 8, pp. 1452–1463, 2006.

[3] J. G. Andrews, S. Buzzi, W. Choi, S. V. Hanly, A. Lozano, A. C. Soong, and J. C. Zhang, "What will 5G be?" *IEEE Journal on selected areas in communications*, vol. 32, no. 6, pp. 1065–1082, 2014.

[4] 3GPP, "Study on integrated access and backhaul," 3rd Generation Partnership Project (3GPP), TR 38.874 (Rel 16), 2019.

[5] L. Tassiulas, "Stability properties of constrained queueing systems and scheduling policies for maximum throughput in multihop radio networks," *IEEE Transactions On Automatic Control*, vol. 31, no. 12, 1992.

[6] S. Gopalam, S. V. Hanly, and P. Whiting, "Distributed and local scheduling algorithms for mmWave integrated access and backhaul," *IEEE/ACM Transactions on Networking*, vol. 30, no. 4, pp. 1749–1764, 2022.

[7] F. P. Kelly, A. K. Maulloo, and D. K. H. Tan, "Rate control for communication networks: shadow prices, proportional fairness and stability," *Journal of the Operational Research society*, vol. 49, no. 3, pp. 237–252, 1998.

[8] S. B. Fred, T. Bonald, A. Proutiere, G. Régnié, and J. W. Roberts, "Statistical bandwidth sharing: a study of congestion at flow level," *ACM SIGCOMM Computer Communication Review*, vol. 31, no. 4, pp. 111–122, 2001.

[9] T. Alpcan and T. Basar, "A utility-based congestion control scheme for internet-style networks with delay," in *IEEE INFOCOM*, vol. 3, 2003, pp. 2039–2048.

[10] J. Mo and J. Walrand, "Fair end-to-end window-based congestion control," *IEEE/ACM Transactions on Networking*, vol. 8, no. 5, pp. 556–567, 2000.

[11] A. Eryilmaz, A. Ozdaglar, D. Shah, and E. Modiano, "Distributed cross-layer algorithms for the optimal control of multihop wireless networks," *IEEE/ACM Transactions on Networking*, vol. 18, no. 2, pp. 638–651, 2010.

[12] M. Gupta, I. P. Roberts, and J. G. Andrews, "System-level analysis of full-duplex self-backhauled millimeter wave networks," *IEEE Transactions on Wireless Communications*, 2022, early access from IEEE Xplore.

[13] Y. Zhang, V. Ramamurthi, Z. Huang, and D. Ghosal, "Co-optimizing performance and fairness using weighted pf scheduling and iab-aware flow control," in *IEEE Wireless Communications and Networking Conference (WCNC)*, 2020, pp. 1–6.

[14] R. Ford, F. Gómez-Cuba, M. Mezzavilla, and S. Rangan, "Dynamic time-domain duplexing for self-backhauled millimeter wave cellular networks," in *IEEE International Conference on Communication Workshop (ICCW)*, 2015, pp. 13–18.

[15] D. Yuan, H.-Y. Lin, J. Widmer, and M. Hollick, "Optimal joint routing and scheduling in millimeter-wave cellular networks," in *IEEE INFOCOM*, 2018, pp. 1205–1213.

[16] M. Eslami Rasekh, D. Guo, and U. Madhow, "Joint routing and resource allocation for millimeter wave picocellular backhaul," *IEEE Transactions on Wireless Communications*, vol. 19, no. 2, pp. 783–794, Feb 2020.

[17] Y. Li, J. Luo, R. Stirling-Gallacher, and G. Caire, "Integrated access and backhaul optimization for millimeter wave heterogeneous networks," *arXiv preprint arXiv:1901.04959v1*, 01 2019.

[18] Y. Niu, C. Gao, Y. Li, L. Su, and D. Jin, "Exploiting multi-hop relaying to overcome blockage in directional mmwave small cells," *Journal of Communications and Networks*, vol. 18, no. 3, pp. 364–374, 2016.

[19] B. Sahoo, C.-H. Yao, and H.-Y. Wei, "Millimeter-wave multi-hop wireless backhauling for 5G cellular networks," in *IEEE Vehicular Technology Conference (VTC Spring)*, June 2017, pp. 1–5.

[20] Y. Yao, H. Tian, G. Nie, H. Wu, and J. Jin, "Multi-path routing based QoS-aware fairness backhaul-access scheduling in mmWave UDN," in *IEEE Annual International Symposium on Personal, Indoor and Mobile Radio Communications (PIMRC)*, Sep. 2018, pp. 1–7.

[21] J. Garcia-Rois, R. Banirazi, F. Gonzalez-Castano, B. Lorenzo, and J. Burguillo, "Delay-aware optimization framework for proportional flow delay differentiation in millimeter-wave backhaul cellular networks," *IEEE Transactions on Communications*, vol. 66, no. 5, pp. 2037–2051, May 2018.

[22] K. Vu, C.-F. Liu, M. Bennis, M. Debbah, and M. Latva-aho, "Path selection and rate allocation in self-backhauled mmWave networks," in *IEEE Wireless Communications and Networking Conference (WCNC)*, April 2018, pp. 1–6.

[23] J. García-Rois, F. Gómez-Cuba, M. R. Akdeniz, F. J. González-Castaño, J. C. Burguillo, S. Rangan, and B. Lorenzo, "On the analysis of scheduling in dynamic duplex multihop mmWave cellular systems," *IEEE Transactions on Wireless Communications*, vol. 14, no. 11, pp. 6028–6042, 2015.

[24] F. Gomez-Cuba and M. Zorzi, "Optimal link scheduling in millimeter wave multi-hop networks with space division multiple access," in *IEEE Information Theory and Applications Workshop (ITA)*, 2016, pp. 1–9.

[25] M. K. Samimi, G. R. MacCartney, S. Sun, and T. S. Rappaport, "28 ghz millimeter-wave ultrawideband small-scale fading models in wireless channels," in *IEEE Vehicular Technology Conference (VTC Spring)*, 2016, pp. 1–6.