On Sample-Path Optimal Dynamic Scheduling for Sum-Queue Minimization in Forests

Srikanth Hariharan* and Ness B. Shroff

***Abstract*—We investigate the problem of minimizing the sum of the queue lengths of all the nodes in a wireless network with a forest topology. Each packet is destined to one of the roots (sinks) of the forest. We consider a time-slotted system, and a primary (or one-hop) interference model. We characterize the existence of causal sample-path optimal scheduling policies for this network topology under this interference model. A causal sample-path optimal scheduling policy is one for which at each time slot, and for any sample-path traffic arrival pattern, the sum of the queue lengths of all the nodes in the network is minimum among *all* policies. We show that such policies exist in restricted forest structures, and that for any other forest structure, there exists a traffic arrival pattern for which no causal sample-path optimal policy can exist. Surprisingly, we show that many forest structures for which such policies exist can be scheduled by converting the structure into an equivalent linear network, and scheduling the equivalent linear network according to the one-hop interference model. The non-existence of such policies in many forest structures underscores the inherent limitation of using sample-path optimality as a performance metric, and necessitates the need to study other (relatively) weaker metrics of delay performance.**

I. Introduction

We investigate the problem of finding causal sample-path optimal scheduling policies for minimizing the sum of the queue lengths of all the nodes in a wireless network with a forest topology. Each packet in the network is destined to one of the roots (sinks) of the forest. We first recall the definition of a sample-path traffic arrival pattern and a causal sample-path optimal scheduling policy for a wireless networks as defined in [10], [9], [4].

Sample-path traffic arrival: Let $\boldsymbol{A}(t)|t \in \{0, 1, 2, ...\}$ be a stochastic process, where $\boldsymbol{A}(t)$ is a random vector (representing traffic arrivals at nodes in the given network) on the probability space $(\Omega, \mathcal{F}, P)$. For any fixed sample point $\omega \in \Omega$, the function $\boldsymbol{A}_\omega(t) : t \rightarrow \boldsymbol{A}(t)$ is called a sample-path of the stochastic process. In other words, considering traffic arrivals as a stochastic process, any sample traffic arrival pattern constitutes a sample-path of the stochastic process.

Sample-path optimal scheduling policy: A sample-path optimal scheduling policy for a wireless network is one for which at each time slot, and for any sample-path traffic arrival pattern, the sum of the queue lengths of all the nodes in the network is minimum among all policies. Further, a causal sample-path optimal scheduling policy is a sample-path optimal scheduling policy that is also causal, i.e., the scheduling decision at any given time slot is independent of future traffic arrivals.

This problem has a number of applications, for instance, in wireless sensor networks with multiple source and destination nodes (sinks) where the sinks require to receive packets in a timely manner from the sensors. Since interference is a critical aspect of wireless networks, it is important to find a transmission schedule such that packets reach the sinks in minimum time. We are interested in minimizing the sum of queue lengths in the system as it can be shown to minimize the long term time average delay experienced by packets in the system.

The convergecasting problem [11] is a special case of the problem considered in this paper in which all the packets in the network are destined to a common sink, i.e., there is only one sink in the network. Delay efficient convergecasting has been well studied in the scheduling literature. Tassiulas et al., [10] first studied the problem of dynamic scheduling for convergecasting in linear networks with the sink at the root of the chain. They showed that for the primary (or one-hop) interference model (where two links that share a node cannot be both active at the same time), for any traffic arrival pattern, any maximal matching policy that gives priority to the link closer to the sink is optimal in the sense that the sum of the queue lengths of all the nodes in the network is minimum at each time slot. This is a very strong result because for any sample-path arrival pattern, this policy is optimal. Also, it is causal as it does not require knowledge of future arrivals. Ji et al., [9] consider small generalized switches with at most four links. They develop a sample-path optimal policy for switches with three links, and a heavy-traffic optimal policy for switches with four links. In [4], Gupta et al., have provided a sample-path optimal policy for a clique wireless network where only one link can transmit at any time, and there are multi-hop flows. Venkataramanan et al., [11] have shown that the policy of giving priority to links closer to the sink is optimal in the large deviations sense, i.e., the rate of decay of the probability that the sum of all the queue lengths exceeds B as $B \rightarrow \infty$ is maximum, even in a general tree topology. In a preliminary version of this paper [6], we have characterized the existence of causal sample-path optimal policies in tree structures. In this paper, we present these results, and also extend our analysis to forest structures. Further, in [8], we have investigated the existence of causal sample-path optimal policies in tree structures under a K-hop interference model, in which no two links that are within K hops

S. Hariharan is with AT&T Labs. Email: srikanth.hariharan@gmail.com

N. B. Shroff is with the Departments of ECE and CSE, The Ohio State University. Email: shroff@ece.osu.edu

* Corresponding author

This work was supported in part by ARO MURI Awards W911NF-07-10376 (SA08-03) and W911NF-08-1-0238, and NSF Awards CNS-0721236, CNS-1012700, and CNS-1065136.

A preliminary version of this paper by S. Hariharan and N. B. Shroff titled "On Optimal Dynamic Scheduling for Sum-Queue Minimization in Trees" appeared in the Proceedings of the 9^{th} IEEE International Symposium on Modeling and Optimization in Mobile, Ad Hoc, and Wireless Networks (WiOpt), 2011 [6].

of each other can be simultaneously activated during a slot.

Apart from the literature considering traffic arrivals, there exist a number of works studying the convergecasting problem in the absence of arrivals. Florens et al., [2] have studied the problem of minimizing the time by which all the packets in a network (with a tree topology) reach the sink, for an one-hop interference model. They propose polynomial time algorithms for this problem. Bermond et al., [1] and Gargano et al., [3] have further studied this problem for disk based communication model, and arbitrary network topologies respectively. In [5], [7], we have studied the convergecasting problem for data aggregation in wireless sensor networks addressing practical constraints such as unreliable links, energy efficiency, and deadline constraints.

The fact that sample-path optimal policies have only been identified so far in very restricted topologies such as linear networks, switches with three links, and networks with a clique-based interference model clearly shows the strength of this metric. Also, we can see that scheduling policies using relatively weaker metrics such as optimality in the large deviations sense, and optimality in the absence of arrivals have been identified for more general networks (trees) under the one-hop interference model. Further, in the absence of arrivals, a number of practical issues in wireless sensor networks have been modeled and investigated for the convergecasting problem.

In this work, we wish to study scheduling in wireless networks with a forest topology for arbitrary traffic arrival patterns. Specifically, we are interested in being able to characterize all possible forest structures for which causal sample-path optimal scheduling policies exist.

Our contributions in this work are the following.

- We first consider forest structures where all the roots of the forest have exactly one common child, and have no other children. The sub-tree rooted at this child can have an arbitrary topology. We show that the policy of giving priority to links closer to the sink minimizes the sum of the queue lengths of all the nodes in the network for every time slot, and for any traffic arrival pattern, by identifying a relationship to a schedule in an equivalent linear network.
- We develop a causal sample-path optimal scheduling policy for single-hop forest networks where all the roots of the forest have exactly one common child, and at most one root has other children.
- We provide a causal sample-path optimal scheduling policy for forest structures with one root (trees) where all but one of the root's children is not a leaf node.
- Surprisingly, we show that for all other forest structures, there exists a traffic arrival pattern such that without having knowledge of future arrivals, there exists no sample-path optimal scheduling policy. *Thus, we completely characterize the existence of causal sample-path optimal scheduling policies in wireless networks with a forest topology under the one-hop interference model*. Given the strength of the sample-path optimality metric, it is an important result to have completely identified the existence of such scheduling policies in a significantly larger class of network topologies than in existing literature.

The rest of this paper is organized as follows. In Section II, we describe the model and notations. In Section III, we discuss forest structures for which causal sample-path optimal scheduling policies exist, and we develop such policies for these structures, and prove their optimality. In Section IV, we show that there exists no causal sample-path optimal scheduling policy in any other forest structure. In Section V, we discuss various metrics of delay efficiency such as evacuation-time optimality, and delay optimality from a large deviations perspective. Finally, we conclude the paper in Section VI.

II. System Model

We model the network as a graph $G(V, E)$, where V is the set of nodes and E is the set of links. The graph G is a forest. We refer to the roots of the forest as sinks. The sinks do not make any transmissions. We assume that the graph is connected, i.e., there is an undirected path from each node to any other node in the network. If there are unconnected components in the network, our results can be immediately extended to such a network by applying the sample-path optimality result to each component independently, i.e., there will exist a causal sample-path optimal policy in such a network if and only if there exists a causal sample-path optimal policy in each of the individual components.

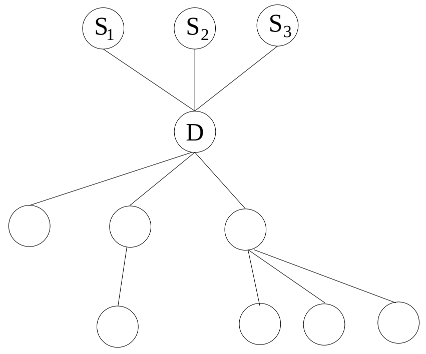
We assume a time-slotted and synchronized system. We consider a one-hop (or node exclusive or primary) interference model where two links that share a node cannot be active at the same time. As in [10], [2], we assume unit capacity links, i.e., a node can at most transmit one packet to its parent during each time slot. Further, each node in the network is equipped with a half-duplex radio transceiver, and therefore a node cannot transmit and receive during the same slot. The external packet arrival pattern at nodes is arbitrary and unknown.

III. Existence of Sample-Path Optimal Policies

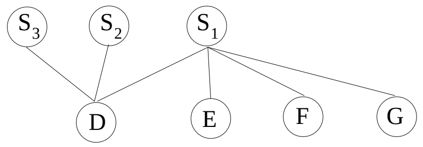
In this section, we consider forest networks for which there exists sample-path optimal scheduling policies, and develop such a policy for these networks. We classify these networks into the following classes, and develop a causal sample-path optimal scheduling policy for each class.

- Class $\mathcal{A}$: This is the class of forest structures where all the roots of the forest have exactly one common child, and have no other children. The sub-tree rooted at this child can have an arbitrary topology.
- Class $\mathcal{B}$: This class is defined as the class of single-hop forest networks where all the roots of the forest have exactly one common child, and at most one root has other children.
- Class $\mathcal{C}$: This is the class of forest structures with one root (trees) where all but one of the root's children is not a leaf node.

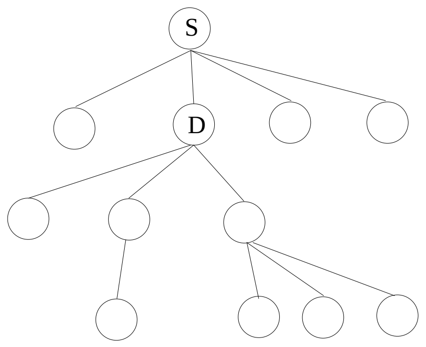
Figure 1 provides examples of Classes $\mathcal{A}$, $\mathcal{B}$, and $\mathcal{C}$. In Figure 1(a), there are three roots S_1, S_2, and S_3 that have a common child D. The roots have no other children, and the structure of the sub-tree rooted at D is arbitrary. Hence, this is a Class $\mathcal{A}$ forest. Figure 1(b) provides an example of a Class $\mathcal{B}$ forest where there are three roots S_1, S_2, and S_3,



(a) Class $\mathcal{A}$



(b) Class $\mathcal{B}$



(c) Class $\mathcal{C}$

Fig. 1. Forest structures for which causal sample-path optimal policies exist

all of which have a common child D. The root S_1 has three other children E, F, and G, while the other roots have D as their only child. Finally, Figure 1(c) is an example of a Class $\mathcal{C}$ forest. Clearly, this structure is a tree with S as the root. S has multiple children but only node D is not a leaf node. The sub-tree rooted at D can be arbitrary.

A. Class $\mathcal{A}$

We first consider Class $\mathcal{A}$ forests. We show that the sample-path optimal policy for this class is to convert this structure into an equivalent linear network [2], and schedule the equivalent linear network according to the one-hop interference model.

We first recall the definition of an equivalent linear network from [2]. Consider a Class $\mathcal{A}$ forest network $G(V,E)$ where V is the set of nodes, E is the set of edges. Suppose S represents an arbitrary root in the forest $G(V,E)$, and each node $u \in V$ has β_u packets during a given time slot. Each packet could be destined to one of the roots of the forest. The equivalent linear network $G(V_l, E_l)$ is defined as follows: $V_l = \{0, 1, ..., N\}$, $E_l = \{(i-1, i), 1 \le i \le N\}$ where $N = \max_{u \in V}(d(S,u))$. $d(S,u)$ represents the distance[1] of node u from the root S. Note that since the network is a Class $\mathcal{A}$ forest, each node in the forest is at the same distance from any of the roots of the forest. Further, each node $j \in V_l$ has α_j packets during the same time slot, where $\alpha_j = \sum_{u \in V: d(S,u)=j} \beta_u$.

Figure 2 gives an example of this transformation. S_1, S_2, and S_3 are the roots (sinks). The farthest node in the forest is 3 hops away from the sinks. Therefore, the equivalent linear network has 3 nodes and the sink node 0. The number of packets at each node is mentioned in the figure. The total number of packets from nodes that are 2 hops away from the sink is 7 (=3+4), and that from nodes that are 3 hops away from the sinks is 9 (=6+1+2). Therefore, the equivalent linear network has 5 packets in node 1, 7 packets in node 2, and 9 packets in node 3.

We show here that the sample-path optimal policy for this class of forest networks is simply to schedule packets in the equivalent linear network according to the sample-path optimal policy for linear networks defined in [10]. In [2], this policy is shown to be *evacuation time optimal* for the class of trees where the root of the tree has only one child but the rest of the tree is arbitrary. We recall the definition of an evacuation time optimal scheduling policy. Consider a wireless network where each node has a certain number of packets initially for a known set of destination nodes. Assume that there are no further packet arrivals in the network. *An evacuation time optimal policy is a scheduling policy such that the time by which the network is evacuated, i.e., the last packet reaches its destination is minimum among all policies.* Clearly, requiring sample-path optimality is a significantly stronger criterion than requiring evacuation time optimality, since sample-path optimality requires optimality at each time slot (and not just optimality at the final time slot) and also optimality for any arbitrary traffic arrival pattern.

[1]Throughout the paper, the "distance" between any two nodes in a forest is simply the number of hops between the nodes. We use the terms "distance" and "hops" interchangeably.

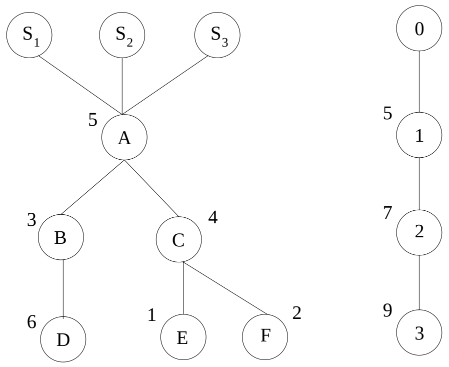


Fig. 2. Equivalent Linear Network

For the reader's convenience, we provide the sample-path optimal policy for linear networks below. We also explain how to convert the schedule for the equivalent linear network into a schedule for the original forest.

Consider a linear network consisting of $N+1$ nodes indexed from 0 to N. Node 0 is the root of the linear network. We have the following notations and definitions (Table I).

For the convergecasting problem, the queue length vector of the linear network evolves as $\boldsymbol{X}(t+1) = \boldsymbol{X}(t) + \boldsymbol{RI}(t+$

TABLE I
DEFINITIONS

1	Activation Set: A set of links that can be simultaneously activated such that no two links interfere with each other according to the one-hop interference model.
2	Activation Vector ($\boldsymbol{I}(t)$): An N-dimensional binary indicator vector $\boldsymbol{i}$ with one element for each link (which is not zero if and only if the link belongs to the activation set).
3	S: Set of all possible activation vectors.
4	$A_i(t)$: Set of exogenous packet arrivals to node i at slot t.
5	$\boldsymbol{A}(t)$: $\boldsymbol{A}(t) = (A_i(t), i = 1, ..., N)$ is the vector of arrivals at all nodes during slot t.
6	$X_i(t)$: Length of the queue of packets at node i by the end of slot t. $X_i(t) \geq 0\ \forall i \in \{1, 2, ..., N\}$.
7	$\boldsymbol{X}(t)$: $\boldsymbol{X}(t) = (X_i(t), i = 1, ..., N)$ is the vector of queue lengths at all nodes at the end of slot t.
8	$\boldsymbol{X}$: The queue length process $\{X(t)\}_{t=0}^{\infty}$.

$1) + \boldsymbol{A}(t+1)$, where $\boldsymbol{R}$ is an $N \times N$ matrix with elements

$$r_{ij} = \begin{cases} 1, & j = i+1 \\ -1, & i = j \\ 0, & \text{otherwise} \end{cases} \tag{1}$$

As explained in [10], the matrix $\boldsymbol{R}$ is a transition matrix representing the change in the queue lengths at each node when a packet is scheduled from one node to another in the network. For instance, suppose that the link $(i+1, i)$ is activated, i.e., a packet is scheduled to be transmitted from node $i+1$ to node i. Then, the queue length at node i increases by 1 ($r_{ij} = 1$ where $j = i+1$), and the queue length at node $i+1$ decreases by 1 ($r_{(i+1)(i+1)} = -1$) because of this scheduling. The queue lengths at all other nodes in the network are not affected by this schedule ($r_{ij} = 0$ for $j \neq i, i+1$).

Note that the above equation is for the equivalent linear network. For the original forest, r_{ij} is the same except that "$j = i+1$" is replaced by "j is a child of i".

Policy π_A: We now define the *stationary policy* π_A which, at slot t, selects the activation vector $\boldsymbol{I}(t) = g_A(\boldsymbol{X}(t-1))$, where $g_A : \mathbb{Z}_+^N \to S$ is defined as follows. Let $\boldsymbol{i} = g_A(\boldsymbol{x})$ and i_j, x_j be the j^{th} elements of vectors $\boldsymbol{i}$ and $\boldsymbol{x}$ respectively. The vector $\boldsymbol{i}$ is defined recursively as follows. $i_1 = 1$, if $x_1 > 0$, and 0, otherwise. For $j = 2, ..., N$, $i_j = 1$, if $x_j > 0$ and $i_{j-1} = 0$. Otherwise, $i_j = 0$.

Policy π_A is clearly causal, and gives priority to packets closer to node 0 in the equivalent linear network.

It has been shown in [10] that π_A is a sample-path optimal scheduling policy for convergecasting in linear networks under the one-hop interference model. However, when we apply this policy to the equivalent linear network that we described earlier, we need to clarify issues regarding transforming the schedule of the equivalent linear network back to a schedule for the original forest network.

- According to policy π_A, any node i in the equivalent linear network can schedule at most one packet during any time slot. This means that among all nodes that are i hops away from node 0 in the original forest, at most one packet will be scheduled. Note that the one-hop interference model allows multiple nodes (at the same distance from the sink) to potentially schedule their transmissions simultaneously if they do not have the same parent. However, policy π_A does not allow such schedules. This implies that even without *Maximal Matching*, this policy is optimal. Note that a *Maximal Matching* policy is one that schedules a set of non-interfering links such that no additional link can be included in the set without interfering with at least one of the existing links, i.e., the set of links scheduled is maximal. It is interesting that even without scheduling additional non-interfering links, this policy is sample-path optimal.
- Suppose that a node i in the equivalent linear network is selected to schedule during a certain slot according to π_A. Consider nodes that are i hops away from node 0 in the original forest that have at least one packet to schedule. One of these nodes can be chosen *arbitrarily* to schedule its packet during that slot. This means that the optimal solution neither depends on the structure of the forest nor the number of packets at each node. For example, in Figure 2, we can arbitrarily choose to schedule one of $\{D, E, F\}$ according to π_A. Further, the policy does not depend on which destination the packet is destined to in the original forest. This is intuitive since a packet is equidistant to any destination for a Class $\mathcal{A}$ forest.
- If a node i in the equivalent linear network is selected to schedule during a certain slot according to π_A, none of the nodes that are $i-1$ hops away from node 0 in the original forest can transmit. Since it is possible to potentially schedule a node that is at distance $i-1$ and a node at distance i simultaneously without interference as long as the node at distance i is not a child of the node at distance $i-1$, it is interesting that even without scheduling such non-interfering links, this policy is optimal. For example, in Figure 2, we can potentially simultaneously schedule B and E. However, this policy does not allow such a schedule because in the equivalent linear network, when node 3 makes a transmission, node 2 cannot make a transmission under the one-hop interference model.

Let $\mathbb{P}$ be the class of all possible activation policies. The proof that π_A is optimal for Class $\mathcal{A}$ forests is similar to Tassiulas's proof for linear networks. Intuitively, the reason that π_A is optimal for such a large class of forests (even though it is not a Maximal Matching policy) is that the links from the sinks' child to the sinks serve as a bottleneck for all the packets in the system. Therefore, even if we allow for a Maximal Matching based schedule, the packets have to ultimately queue up at the sinks' child, and get transmitted one after another.

Theorem 1. *Consider the evolution of the system (Class $\mathcal{A}$ forest) under policy π_A and an arbitrary policy $\pi \in \mathbb{P}$. Let $\boldsymbol{X}$, $\boldsymbol{X}^0$ be the queue length processes under π and π_A respectively when the system starts from the same initial state[2] under both policies. For all $t = 0, 1, ...$ we have*

$$\sum_{i \in V} X_i^0(t) \leq \sum_{i \in V} X_i(t) \text{ a.s.} \tag{2}$$

We first provide some definitions and lemmas before going into the proof of the theorem.

Definition: Let $\boldsymbol{X}$, $\boldsymbol{Y}$ be the queue length processes when

[2]Throughout the paper, the "state" of the system refers to the queue lengths of all the nodes in the system.

the initial queue length vectors are $\boldsymbol{X}(0) = \boldsymbol{x}$, $\boldsymbol{Y}(0) = \boldsymbol{y}$ respectively, there are no exogenous arrivals, and policy π_A schedules link activations. We say that the vectors $\boldsymbol{x}$ and $\boldsymbol{y}$ are related with the partial ordering $\prec$, and we write $\boldsymbol{x} \prec \boldsymbol{y}$, if for all $t = 0, 1, ...,$ we have $l(\boldsymbol{X}(t)) \leq l(\boldsymbol{Y}(t))$, where the function $l(\cdot)$ represents the total number of packets in the system for a given state. To be precise, for any system state $\boldsymbol{z} = \{z_i, i \in V\}$, $l(\boldsymbol{z}) = \sum_{i \in V} z_i$.

To each state $\boldsymbol{x}$ we define the *departure times* $t_i^{\boldsymbol{x}}$, $i = 1, ..., l(\boldsymbol{x})$ and the *positions* $d_i^{\boldsymbol{x}}$, $i = 1, ..., l(\boldsymbol{x})$ as follows.

Definition: Assume that the system is initially in state $\boldsymbol{x}$, there are no exogenous arrivals, and policy π_A schedules link activations. Let $\{\boldsymbol{X}(t)\}_{t=0}^{\infty}$ be the corresponding queue length process. Let S represent any of the roots of the forest. The time $t_i^{\boldsymbol{x}}$ is defined as

$$t_i^{\boldsymbol{x}} = \min\{t : t > 0, l(\boldsymbol{X}(t)) \leq l(\boldsymbol{x}) - i\},\ i = 1, ..., l(\boldsymbol{x}), \quad (3)$$

and the position $d_i^{\boldsymbol{x}}$ is defined as

$$d_i^{\boldsymbol{x}} = \max\{j+1 : j \geq 0, \sum_{n \in V : d(S,n) \leq j} x_n < i\},\ i = 1, ..., l(\boldsymbol{x}). \quad (4)$$

Note that the definitions have been appropriately modified for our topology. The corresponding definitions for the equivalent linear network will be exactly the same as in Definition 3.2 in [10]. Let us now index the packets by an index i that denotes the order in which the packets reach their destinations when the system is in state $\boldsymbol{x}$ at $t = 0$, π_A schedules link activation, and there are no exogenous arrivals. The departure time $t_i^{\boldsymbol{x}}$ is the slot by the end of which packet i reaches its destination (node 0 in the equivalent linear network), and the position $d_i^{\boldsymbol{x}}$ is the distance of the node (from node 0 in the equivalent linear network) at which packet i was residing at $t = 0$. In the equivalent linear network, if $d_i^{\boldsymbol{x}} = k$, then packet i was residing at node k at $t = 0$.

We now show that the departure times and the positions for our topology are related in the same manner as in Equation (3.4) in [10].

Lemma 1. *For Class $\mathcal{A}$ forests, for all states $\boldsymbol{x}$ we have*

$$t_i^{\boldsymbol{x}} = \begin{cases} d_i^{\boldsymbol{x}} & i = 1 \\ i & d_i^{\boldsymbol{x}} = 1 \\ \max\{t_{i-1}^{\boldsymbol{x}} + 2, d_i^{\boldsymbol{x}}\} & i > 1, d_i^{\boldsymbol{x}} > 1 \end{cases} \quad (5)$$

Proof: Before we go into the details, we recall that a packet that is destined to one of the roots of the Class $\mathcal{A}$ forest is destined to node 0 in the equivalent linear network. Let S represent any of the roots of the Class $\mathcal{A}$ forest. Further, node 1 in the equivalent linear network corresponds to the common child of all the roots in the Class $\mathcal{A}$ forest. Therefore, in order to reach any destination (one of the roots), a packet needs to traverse through node 1.

Consider the system operated under policy π_A, with initial state $\boldsymbol{x}$ and without arrivals. The time taken by the first packet to exit the system is simply the distance of the node at which it was residing at $t = 0$ to the destination (node 0 in the equivalent linear network) because it gets forwarded by one hop during each time slot. Therefore, $t_1^{\boldsymbol{x}} = d_1^{\boldsymbol{x}}$.

For a packet i such that $d_i^{\boldsymbol{x}} = 1$, the time taken by this packet to leave the system is i since at each slot one packet will be forwarded from node 1 (in the equivalent linear network, and hence in the original forest) to its destination until the time that node 1 has no more packets to send. Therefore, in this case, the packet i will reach node 0 in the equivalent linear network at the end of slot i. Therefore, if $d_i^{\boldsymbol{x}} = 1$, $t_i^{\boldsymbol{x}} = i$.

If $i > 1$ and $d_i^{\boldsymbol{x}} > 1$, we distinguish the following cases.

Case 1: $d_i^{\boldsymbol{x}} - t_{i-1}^{\boldsymbol{x}} \geq 2$.

At any slot $t < t_{i-1}^{\boldsymbol{x}}$, the packet $i-1$ should reside in a node j in the original forest such that $d(S,j) \leq t_{i-1}^{\boldsymbol{x}} - t$ because it should reach the destination in $t_{i-1}^{\boldsymbol{x}} - t$ slots, and cannot be forwarded faster than one hop during each slot. Also, at time t, the packet i should reside in a node m such that $d(S,m) \geq d_i^{\boldsymbol{x}} - t$ since it cannot move faster towards the destination than one hop per slot. Therefore we have $d(S,m) \geq d_i^{\boldsymbol{x}} - t \geq t_{i-1}^{\boldsymbol{x}} - t + 2 \geq d(S,j) + 2$. This implies that packet $i-1$ will be, at each slot t, at least two hops closer to the destination than packet i in both the original forest as well as the equivalent linear network. Therefore packet i will be the first packet in its queue (according to our convention), and all the nodes in the forest that are one hop closer to the destination than the node at which packet i currently is have no packets in their respective queues. Therefore, packet i will be forwarded by one node towards the destination at each slot. Hence, packet i will reach the destination by the end of slot $d_i^{\boldsymbol{x}}$, i.e., $t_i^{\boldsymbol{x}} = d_i^{\boldsymbol{x}}$.

Case 2: $d_i^{\boldsymbol{x}} - t_{i-1}^{\boldsymbol{x}} \leq 1$.

If $i > 1$ and $d_i^{\boldsymbol{x}} > 1$, then $t_i^{\boldsymbol{x}} \geq t_{i-1}^{\boldsymbol{x}} + 2$. This is because any packet which is not residing in node 1 in the original forest (or the equivalent linear network) at $t = 0$, can reach node 1 only when there are no packets left to schedule in node 1, since node 1 is activated otherwise, and because of the one-hop interference model[3]. Hence, during the slot at which $i-1$ leaves the system, packet i will be in node 2 in the equivalent linear network (corresponding to one of the children of node 1 in the original forest) or further away from its destination, and therefore it requires at least two additional slots in order to reach its destination.

We now show that $t_i^{\boldsymbol{x}} = t_{i-1}^{\boldsymbol{x}} + 2$. If packet i is forwarded towards its destination by one node at each slot then it will reach its destination by slot $d_i^{\boldsymbol{x}}$. However, this is impossible since $d_i^{\boldsymbol{x}} - t_{i-1}^{\boldsymbol{x}} \leq 1$, and we need $t_i^{\boldsymbol{x}} \geq t_{i-1}^{\boldsymbol{x}} + 2$. This means that at some slot, packet i is not forwarded from its node (say node k). Suppose that packet $i-1$ was residing at node j during this slot. Then we must either have $d(S,j) = d(S,k)$, or $d(S,j) = d(S,k) - 1$, i.e., in the equivalent linear network packet $i-1$ is either in the same node with i or in the node in front of i towards the destination. Therefore, at the slot at which i was not forwarded and at all subsequent slots until the time packet $i-1$ leaves the system, packets i and $i-1$ cannot be in two nodes m, n such that $d(S,m) - d(S,n) > 2$. Therefore, two slots after the time packet $i-1$ reaches node 0 in the equivalent linear

[3]Note that this is true only when the roots have only one common child, and have no other children. In a general forest, simultaneous transmissions are possible among two nodes at distance one to their respective (different) roots. Simultaneous transmissions are also possible among a node at distance one from one of the roots' children, and a node at distance two in another branch belonging to the same root. The fact that such transmissions are not possible in Class $\mathcal{A}$ forests is one of the most important reasons why all the proofs in [10] works for this topology. Node 1 serves as a bottleneck.

network, packet i also reaches node 0. Thus, $t_i^{\boldsymbol{x}} = t_{i-1}^{\boldsymbol{x}} + 2$. ■

We now recall Lemma 3.2 in [10] below. This lemma is independent of the network topology, and is a property of the sample-path optimality metric.

Lemma 2. *For any two vectors $\boldsymbol{x}$ and $\boldsymbol{y}$, we have $\boldsymbol{x} \prec \boldsymbol{y}$ if and only if $t_i^{\boldsymbol{x}} \leq t_{i+k}^{\boldsymbol{y}},\ i = 1, ..., l(\boldsymbol{x})$, where $k = l(\boldsymbol{y}) - l(\boldsymbol{x})$.*

Lemma 2 implies that if the initial difference in the sum of the queue lengths between states $\boldsymbol{x}$ and $\boldsymbol{y}$ is k, then $\boldsymbol{x} \prec \boldsymbol{y}$ if and only if for any i, the time for the i^{th} packet to leave the system according to state $\boldsymbol{x}$ is no greater than the time for the $(i+k)^{th}$ packet to leave the system according to state $\boldsymbol{y}$.

We now show that if there are no exogenous arrivals, policy π_A minimizes the sum of the queue lengths of all the nodes in the system for each time slot.

Lemma 3. *If we have $\boldsymbol{x} \prec \boldsymbol{y}$, and $\boldsymbol{i}$ is an arbitrary activation vector, then for $\boldsymbol{u} = \boldsymbol{x} + \boldsymbol{R}g_A(\boldsymbol{x})$ and $\boldsymbol{z} = \boldsymbol{y} + \boldsymbol{R}\boldsymbol{i}$, we have $\boldsymbol{u} \prec \boldsymbol{z}$.*

Proof: We show that for all $i = 1, ..., l(\boldsymbol{x})$ we have $t_i^{\boldsymbol{u}} \leq t_{i+l(\boldsymbol{z})-l(\boldsymbol{u})}^{\boldsymbol{z}}$ and thus from the previous lemma, we must have $\boldsymbol{u} \prec \boldsymbol{z}$.

Let $l(\boldsymbol{y}) - l(\boldsymbol{x}) = k$. We have four cases.

Case 1: $l(\boldsymbol{u}) = l(\boldsymbol{x})$, $l(\boldsymbol{z}) = l(\boldsymbol{y})$.

In this case, we need to show that $t_i^{\boldsymbol{u}} \leq t_{i+k}^{\boldsymbol{z}},\ \forall\ i = 1, ..., l(\boldsymbol{u})$.

Since $\boldsymbol{u}$ results from applying policy π_A, from the definition of departure times, it immediately follows that

$$t_i^{\boldsymbol{u}} = t_i^{\boldsymbol{x}} - 1. \tag{6}$$

We now show by induction on i that

$$t_i^{\boldsymbol{y}} \geq t_i^{\boldsymbol{z}} \geq t_i^{\boldsymbol{y}} - 1. \tag{7}$$

$i = 1$: We have $t_1^{\boldsymbol{y}} = d_1^{\boldsymbol{y}}$, and $d_1^{\boldsymbol{y}} \geq d_1^{\boldsymbol{z}} \geq d_1^{\boldsymbol{y}} - 1$ for the first packet. The first equation follows from Lemma 1, and the second follows from the fact that the first packet cannot travel more than one hop in one slot. Therefore, we have $t_1^{\boldsymbol{z}} = d_1^{\boldsymbol{z}} \geq d_1^{\boldsymbol{y}} - 1 = t_1^{\boldsymbol{y}} - 1$, and $t_1^{\boldsymbol{z}} \leq d_1^{\boldsymbol{y}} = t_1^{\boldsymbol{y}}$. Thus, the result holds for $i = 1$.

By the induction hypothesis, assume that the result holds for some packet i.

$i+1$: If $d_{i+1}^{\boldsymbol{z}} = 1$, then $t_{i+1}^{\boldsymbol{z}} = i+1$. $t_{i+1}^{\boldsymbol{y}} = i+2$ if the packet $i+1$ was in node 2 in the equivalent linear network and activated by activation vector $\boldsymbol{i}$ or $t_{i+1}^{\boldsymbol{y}} = i+1$ if the packet $i+1$ was in node 1 and it was not scheduled by activation vector $\boldsymbol{i}$. Therefore, $t_{i+1}^{\boldsymbol{z}} = t_{i+1}^{\boldsymbol{y}}$ or $t_{i+1}^{\boldsymbol{z}} = t_{i+1}^{\boldsymbol{y}} - 1$. Thus the result holds when $d_{i+1}^{\boldsymbol{z}} = 1$.

If $d_{i+1}^{\boldsymbol{z}} > 1$, then $t_{i+1}^{\boldsymbol{z}} = \max\{t_i^{\boldsymbol{z}} + 2, d_{i+1}^{\boldsymbol{z}}\}$. We show that either $d_{i+1}^{\boldsymbol{z}} = d_{i+1}^{\boldsymbol{y}} - 1$, or $d_{i+1}^{\boldsymbol{z}} = d_{i+1}^{\boldsymbol{y}}$. Suppose that packet $i+1$ in state $\boldsymbol{y}$ is at distance d from the roots of the forest. Even if the activation vector $\boldsymbol{i}$ schedules multiple packets at the same distance to the roots, the packet $i+1$ in state $\boldsymbol{z}$ can either be at a node at distance $d-1$ from the roots, or at a node at distance d from the roots since any packet can at most reach one hop closer to the sink during a single time slot. Thus, $d_{i+1}^{\boldsymbol{z}} = d_{i+1}^{\boldsymbol{y}} - 1$, or $d_{i+1}^{\boldsymbol{z}} = d_{i+1}^{\boldsymbol{y}}$.

Now, we have $t_{i+1}^{\boldsymbol{y}} - 1 = \max\{t_i^{\boldsymbol{y}} + 2, d_{i+1}^{\boldsymbol{y}}\} - 1 = \max\{t_i^{\boldsymbol{y}} - 1 + 2, d_{i+1}^{\boldsymbol{y}} - 1\} \leq \max\{t_i^{\boldsymbol{z}} + 2, d_{i+1}^{\boldsymbol{z}}\} = t_{i+1}^{\boldsymbol{z}} \leq \max\{t_i^{\boldsymbol{y}} + 2, d_{i+1}^{\boldsymbol{y}}\} = t_{i+1}^{\boldsymbol{y}}$, where the third relation follows because $t_i^{\boldsymbol{z}} \geq t_i^{\boldsymbol{y}} - 1$ and $d_{i+1}^{\boldsymbol{z}} \geq d_{i+1}^{\boldsymbol{y}} - 1$, and the fourth relation holds because $t_i^{\boldsymbol{z}} \leq t_i^{\boldsymbol{y}}$ and $d_{i+1}^{\boldsymbol{z}} \leq d_{i+1}^{\boldsymbol{y}}$.

Therefore by induction, the relation (7) holds. From the relations (6), (7), and the fact that $\boldsymbol{x} \prec \boldsymbol{y}$, it follows that $t_i^{\boldsymbol{u}} \leq t_{i+k}^{\boldsymbol{z}},\ \forall\ i = 1, ..., l(\boldsymbol{u})$.

Case 2: $l(\boldsymbol{u}) = l(\boldsymbol{x}) - 1$, $l(\boldsymbol{z}) = l(\boldsymbol{y})$.

In this case, we need to show that $t_i^{\boldsymbol{u}} \leq t_{i+k+1}^{\boldsymbol{z}},\ \forall\ i = 1, ..., l(\boldsymbol{u})$.

Since one packet exits the system according to policy π_A, the $i+1^{th}$ packet in the previous slot now becomes the i^{th} packet. Therefore,

$$t_i^{\boldsymbol{u}} = t_{i+1}^{\boldsymbol{x}}. \tag{8}$$

For $\boldsymbol{z}$, the situation is identical to that of Case 1. Therefore, the relation (7) holds. Therefore, it follows that $t_i^{\boldsymbol{u}} \leq t_{i+k+1}^{\boldsymbol{z}}$, $\forall\ i = 1, ..., l(\boldsymbol{u})$.

Case 3: $l(\boldsymbol{u}) = l(\boldsymbol{x}) - 1$, $l(\boldsymbol{z}) = l(\boldsymbol{y}) - 1$.

In this case, we need to show that $t_i^{\boldsymbol{u}} \leq t_{i+k}^{\boldsymbol{z}},\ \forall\ i = 1, ..., l(\boldsymbol{u})$.

From Case 2 for $\boldsymbol{u}$, we have $t_i^{\boldsymbol{u}} = t_{i+1}^{\boldsymbol{x}}$.

For $\boldsymbol{z}$, we now show by induction that

$$t_{i+1}^{\boldsymbol{y}} \geq t_i^{\boldsymbol{z}} \geq t_{i+1}^{\boldsymbol{y}} - 1. \tag{9}$$

$i = 1$: We have $t_1^{\boldsymbol{z}} = d_1^{\boldsymbol{z}}$. Also, $t_2^{\boldsymbol{y}} \geq d_2^{\boldsymbol{y}}$. If $t_2^{\boldsymbol{y}} = d_2^{\boldsymbol{y}}$, then $t_1^{\boldsymbol{z}} = d_1^{\boldsymbol{z}} \geq d_2^{\boldsymbol{y}} - 1 \geq t_2^{\boldsymbol{y}} - 1$, and $t_1^{\boldsymbol{z}} = d_1^{\boldsymbol{z}} \leq d_2^{\boldsymbol{y}} \leq t_2^{\boldsymbol{y}}$. This is because $d_2^{\boldsymbol{y}} - 1 \leq d_1^{\boldsymbol{z}} \leq d_2^{\boldsymbol{y}}$ (since the second packet according to state $\boldsymbol{y}$ could have at most moved one hop closer to its destination). Therefore, the result holds in this case. On the other hand, it is also possible that $t_2^{\boldsymbol{y}} = d_2^{\boldsymbol{y}} + 1$ if the second packet resided at node 2 at the previous slot in the equivalent linear network. Since a packet left the system during this slot (since $l(\boldsymbol{z}) = l(\boldsymbol{y}) - 1$), the second packet still remains at node 2. In this case, $t_1^{\boldsymbol{z}} = d_1^{\boldsymbol{z}} = d_2^{\boldsymbol{y}} = t_2^{\boldsymbol{y}} - 1$. Therefore, the relation (9) holds for the first packet in state $\boldsymbol{z}$.

Assume that it holds for some packet i by the induction hypothesis.

$i+1$: If $d_{i+1}^{\boldsymbol{z}} = 1$, then $t_{i+1}^{\boldsymbol{z}} = i+1$, and $t_{i+2}^{\boldsymbol{y}} = i+2$. Hence, $t_{i+2}^{\boldsymbol{y}} \geq t_{i+1}^{\boldsymbol{z}} \geq t_{i+2}^{\boldsymbol{y}} - 1$.

If $d_{i+1}^{\boldsymbol{z}} > 1$, then by Lemma 1, we have $t_{i+1}^{\boldsymbol{z}} = \max\{t_i^{\boldsymbol{z}} + 2, d_{i+1}^{\boldsymbol{z}}\}$. We now verify (9) for both values of $t_{i+1}^{\boldsymbol{z}}$.

If $t_{i+1}^{\boldsymbol{z}} = d_{i+1}^{\boldsymbol{z}}$, then it follows that $t_{i+1}^{\boldsymbol{z}} = d_{i+1}^{\boldsymbol{z}} \leq d_{i+2}^{\boldsymbol{y}} \leq t_{i+2}^{\boldsymbol{y}}$.

If $t_{i+1}^{\boldsymbol{z}} = t_i^{\boldsymbol{z}} + 2$, then $t_{i+1}^{\boldsymbol{z}} = t_i^{\boldsymbol{z}} + 2 \leq t_{i+1}^{\boldsymbol{y}} + 2 \leq t_{i+2}^{\boldsymbol{y}}$, where the second relation follows by the induction hypothesis.

If $t_{i+2}^{\boldsymbol{y}} = d_{i+2}^{\boldsymbol{y}}$, then $t_{i+2}^{\boldsymbol{y}} = d_{i+2}^{\boldsymbol{y}} \leq d_{i+1}^{\boldsymbol{z}} + 1 \leq t_{i+1}^{\boldsymbol{z}} + 1$, since the packet $i+2$ in state $\boldsymbol{y}$ can move at most one hop closer to the destination according to activation vector $\boldsymbol{i}$.

If $t_{i+2}^{\boldsymbol{y}} = t_{i+1}^{\boldsymbol{y}} + 2$, then $t_{i+1}^{\boldsymbol{z}} \geq t_i^{\boldsymbol{z}} + 2 \geq t_{i+1}^{\boldsymbol{y}} + 2 - 1 = t_{i+2}^{\boldsymbol{y}} - 1$, where the second inequality follows by the induction hypothesis.

From the four relations above, the relation (9) holds for $i+1$. Therefore, by induction, it holds for all i.

The relations (8) and (9) imply that $t_i^{\boldsymbol{u}} \leq t_{i+k}^{\boldsymbol{z}},\ \forall\ i = 1, ..., l(\boldsymbol{u})$.

Case 4: $l(\boldsymbol{u}) = l(\boldsymbol{x})$, $l(\boldsymbol{z}) = l(\boldsymbol{y}) - 1$.

In this case, we need to show that $\forall i$, $t_i^{\boldsymbol{u}} \le t_{i+k-1}^{\boldsymbol{z}}$.

The case for $\boldsymbol{u}$ is identical to that in Case 1, and the case for $\boldsymbol{z}$ is identical to that in Case 3. Hence, the result follows.

Thus, we have shown that $\boldsymbol{u} \prec \boldsymbol{z}$. ∎

We now show that the ordering $\prec$ is preserved even after a packet arrives at any node in the network. To be precise, let $\boldsymbol{e}_j$ be the vector which has all its elements equal to zero except for the element j which is 1. Then we have the following.

Lemma 4. *If we have $\boldsymbol{x} \prec \boldsymbol{y}$, then for all $j \in V$, we also have $\boldsymbol{x} + \boldsymbol{e}_j \prec \boldsymbol{y} + \boldsymbol{e}_j$.*

Proof: First note that if a packet arrives at a node j in the forest at a distance r from any of the roots of the forest, then it arrives at node r in the equivalent linear network.

Let $\boldsymbol{u} = \boldsymbol{x} + \boldsymbol{e}_j$, and $\boldsymbol{z} = \boldsymbol{y} + \boldsymbol{e}_j$. Since $\boldsymbol{x} \prec \boldsymbol{y}$, $l(\boldsymbol{y}) - l(\boldsymbol{x}) = k \ge 0$. Thus, $l(\boldsymbol{z}) - l(\boldsymbol{u}) = k$.

We need to show that $\forall i = 1, 2, ..., l(\boldsymbol{x}) + 1$,

$$t_i^{\boldsymbol{u}} \le t_{i+k}^{\boldsymbol{z}}. \tag{10}$$

Case 1: Packet i in state $\boldsymbol{u}$ leaves before the newly arrived packet in state $\boldsymbol{u}$, and packet $i + k$ in state $\boldsymbol{z}$ leaves before the newly arrived packet in state $\boldsymbol{z}$. Further, neither i nor $i + k$ is the newly arrived packet.

For all such packets, $t_i^{\boldsymbol{u}} = t_i^{\boldsymbol{x}}$, and $t_{i+k}^{\boldsymbol{z}} = t_{i+k}^{\boldsymbol{y}}$ because the transmission schedules of these packets are not affected by the arrival of a packet at node j. Therefore, the relation (10) holds for this case.

Case 2: Packet i in state $\boldsymbol{u}$ is either the newly arrived packet or leaves after the newly arrived packet, and packet $i + k$ in state $\boldsymbol{z}$ leaves before the newly arrived packet, and is not the newly arrived packet.

We have $d_i^{\boldsymbol{u}} \le d_i^{\boldsymbol{x}}$ $\forall i = 1, ..., l(\boldsymbol{x})$ because if the packet i in state $\boldsymbol{u}$ leaves before the newly arrived packet, $d_i^{\boldsymbol{u}} = d_i^{\boldsymbol{x}}$, and if i is either the newly arrived packet or leaves after the newly arrived packet, $d_i^{\boldsymbol{u}} = d_{i-1}^{\boldsymbol{x}} \le d_i^{\boldsymbol{x}}$.

We want to show now that $t_i^{\boldsymbol{u}} \le t_i^{\boldsymbol{x}}$. We do this by induction.

$i = 1$: We have $t_1^{\boldsymbol{u}} = d_1^{\boldsymbol{u}} \le d_1^{\boldsymbol{x}} = t_1^{\boldsymbol{x}}$. Thus, it holds for $i = 1$.

Assume that $t_i^{\boldsymbol{u}} \le t_i^{\boldsymbol{x}}$ for some packet i by the induction hypothesis.

$i + 1$: If $t_{i+1}^{\boldsymbol{u}} = d_{i+1}^{\boldsymbol{u}}$ and $t_{i+1}^{\boldsymbol{x}} = d_{i+1}^{\boldsymbol{x}}$, then since $d_{i+1}^{\boldsymbol{u}} \le d_{i+1}^{\boldsymbol{x}}$, we have $t_{i+1}^{\boldsymbol{u}} \le t_{i+1}^{\boldsymbol{x}}$.

If $t_{i+1}^{\boldsymbol{u}} = t_i^{\boldsymbol{u}} + 2$ and $t_{i+1}^{\boldsymbol{x}} = t_i^{\boldsymbol{x}} + 2$, then it immediately follows that $t_{i+1}^{\boldsymbol{u}} \le t_{i+1}^{\boldsymbol{x}}$ since $t_i^{\boldsymbol{u}} \le t_i^{\boldsymbol{x}}$ by the induction hypothesis.

If $t_{i+1}^{\boldsymbol{u}} = d_{i+1}^{\boldsymbol{u}}$ and $t_{i+1}^{\boldsymbol{x}} = t_i^{\boldsymbol{x}} + 2$, then we have $t_i^{\boldsymbol{x}} + 2 \ge d_{i+1}^{\boldsymbol{x}} \ge d_{i+1}^{\boldsymbol{u}} = t_{i+1}^{\boldsymbol{u}}$.

If $t_{i+1}^{\boldsymbol{u}} = t_i^{\boldsymbol{u}} + 2$ and $t_{i+1}^{\boldsymbol{x}} = d_{i+1}^{\boldsymbol{x}}$, then we have $d_{i+1}^{\boldsymbol{x}} \ge t_i^{\boldsymbol{x}} + 2 \ge t_i^{\boldsymbol{u}} + 2 = t_{i+1}^{\boldsymbol{u}}$.

Thus, the result holds for $i+1$. Hence, by induction, $t_i^{\boldsymbol{u}} \le t_i^{\boldsymbol{x}}$ $\forall i$.

Also, from Case 1, for $\boldsymbol{z}$, $t_{i+k}^{\boldsymbol{z}} = t_{i+k}^{\boldsymbol{y}}$ when packet $i + k$ is one that leaves before the newly arrived packet in state $\boldsymbol{z}$.

Therefore, we have $t_i^{\boldsymbol{u}} \le t_i^{\boldsymbol{x}} \le t_{i+k}^{\boldsymbol{y}} = t_{i+k}^{\boldsymbol{z}}$. Hence, $t_i^{\boldsymbol{u}} \le t_{i+k}^{\boldsymbol{z}}$.

Case 3: Packet i in state $\boldsymbol{u}$ is either the newly arrived packet or leaves before the newly arrived packet, and packet $i + k$ in state $\boldsymbol{z}$ is either the newly arrived packet or leaves after the newly arrived packet.

For this case, we give a proof by contradiction.

Suppose that for some i, $t_i^{\boldsymbol{u}} > t_{i+k}^{\boldsymbol{z}}$.

Since $i + k$ in state $\boldsymbol{z}$ is either the newly arrived packet or leaves after the newly arrived packet, and i in state $\boldsymbol{u}$ is either the newly arrived packet or leaves before the newly arrived packet, we have $t_{i+k}^{\boldsymbol{z}} \ge d_{i+k}^{\boldsymbol{z}} \ge d_i^{\boldsymbol{u}}$, since packets closer to the roots leave the system before packets farther away from the roots.

Hence, $t_i^{\boldsymbol{u}} > t_{i+k}^{\boldsymbol{z}}$ means that $t_i^{\boldsymbol{u}} > d_i^{\boldsymbol{u}}$. Therefore, $t_i^{\boldsymbol{u}} = t_{i-1}^{\boldsymbol{u}} + 2$. So $t_{i-1}^{\boldsymbol{u}} + 2 > t_{i+k}^{\boldsymbol{z}} \ge t_{i+k-1}^{\boldsymbol{z}} + 2$. This implies that $t_{i-1}^{\boldsymbol{u}} > t_{i+k-1}^{\boldsymbol{z}}$.

Iteratively substitute i by $i - 1$ until either $i = 1$ in state $\boldsymbol{u}$ or $i + k$ is a packet that leaves before the newly arrived packet in state $\boldsymbol{z}$. If $i = 1$ in state $\boldsymbol{u}$, then $t_1^{\boldsymbol{u}} = d_1^{\boldsymbol{u}}$. However, this contradicts $t_i^{\boldsymbol{u}} > d_i^{\boldsymbol{u}}$ $\forall i$. If $i + k$ in state $\boldsymbol{z}$ is a packet that leaves before the newly arrived packet, then $t_{i+k}^{\boldsymbol{z}} = t_{i+k}^{\boldsymbol{y}}$ and $t_i^{\boldsymbol{u}} \le t_i^{\boldsymbol{x}}$. However, $t_{i+k}^{\boldsymbol{z}} < t_i^{\boldsymbol{u}}$ then contradicts the fact that $\boldsymbol{x} \prec \boldsymbol{y}$.

Hence $t_i^{\boldsymbol{u}} \le t_{i+k}^{\boldsymbol{z}}$ by contradiction.

Case 4: Packet i in state $\boldsymbol{u}$ leaves the system after the newly arrived packet, and packet $i + k$ in state $\boldsymbol{z}$ is either the newly arrived packet or leaves the system after the newly arrived packet.

For packets in state $\boldsymbol{u}$ that leave the system after the newly arrived packet, we have $d_i^{\boldsymbol{u}} = d_{i-1}^{\boldsymbol{x}}$. Similarly, for packets in state $\boldsymbol{z}$ that leave the system after the newly arrived packet, we have $d_{i+k}^{\boldsymbol{z}} = d_{i+k-1}^{\boldsymbol{y}}$.

Suppose that the new packet is the m^{th} packet to leave the system according to state $\boldsymbol{u}$, and the n^{th} packet to leave the system according to state $\boldsymbol{z}$.

If $i + k = n < l(\boldsymbol{z})$, $d_n^{\boldsymbol{z}} \le d_n^{\boldsymbol{y}}$, and $t_{n-1}^{\boldsymbol{z}} = t_{n-1}^{\boldsymbol{y}}$. So $t_n^{\boldsymbol{z}} = \max\{t_{n-1}^{\boldsymbol{z}} + 2, d_n^{\boldsymbol{z}}\} \le \max\{t_{n-1}^{\boldsymbol{y}} + 2, d_n^{\boldsymbol{y}}\} = t_n^{\boldsymbol{y}}$. Therefore, $t_n^{\boldsymbol{z}} \le t_n^{\boldsymbol{y}}$.

We now show that $t_{i+k}^{\boldsymbol{z}} \ge t_{i+k-1}^{\boldsymbol{y}}$ when $i + k \ge n$.

First, consider the new packet $i + k = n$: Since $t_n^{\boldsymbol{z}} \ge t_{n-1}^{\boldsymbol{z}} = t_{n-1}^{\boldsymbol{y}}$, we have $t_n^{\boldsymbol{z}} \ge t_{n-1}^{\boldsymbol{y}}$. Thus the result holds for $i + k = n$.

Assume that the result holds for a packet $i + k = l > n$ in state $\boldsymbol{z}$, i.e., $t_l^{\boldsymbol{z}} \ge t_{l-1}^{\boldsymbol{y}}$.

Consider $i + k = l + 1$: We have $t_{l+1}^{\boldsymbol{z}} = \max\{t_l^{\boldsymbol{z}} + 2, d_{l+1}^{\boldsymbol{z}}\} \ge \max\{t_{l-1}^{\boldsymbol{y}} + 2, d_l^{\boldsymbol{y}}\} = t_l^{\boldsymbol{y}}$, since $d_{l+1}^{\boldsymbol{z}} = d_l^{\boldsymbol{y}}$ and $t_l^{\boldsymbol{z}} \ge t_{l-1}^{\boldsymbol{y}}$.

Thus the result holds for $l + 1$.

Therefore, by induction, $t_{i+k}^{\boldsymbol{z}} \ge t_{i+k-1}^{\boldsymbol{y}}$ $\forall$ $i + k \ge n$.

We now prove (10) for Case 4.

If $d_{i+k}^{\boldsymbol{z}} = 1$, then $i + k = n$ is the new packet according to state $\boldsymbol{z}$ (because the new packet will be placed at the end of the queue at node 1 in the equivalent linear network). Suppose that in state $\boldsymbol{u}$ we have $i = n - k > m$. We need to show that $t_{n-k}^{\boldsymbol{u}} \le t_n^{\boldsymbol{z}}$. We note that the arrival of the new packet at node 1 in the equivalent linear network at most increases the time for packets that leave after the new packet by one slot, i.e., $t_{n-k}^{\boldsymbol{u}} \le t_{n-k-1}^{\boldsymbol{x}} + 1$. We now have $t_{n-k}^{\boldsymbol{u}} \le t_{n-k-1}^{\boldsymbol{x}} + 1 \le t_{n-1}^{\boldsymbol{y}} + 1 = t_n^{\boldsymbol{z}}$. Hence, (10) holds.

We prove the other cases by contradiction. Suppose that $t_i^{\boldsymbol{u}} > t_{i+k}^{\boldsymbol{z}}$ for some i.

If $t_i^{\boldsymbol{u}} = d_i^{\boldsymbol{u}}$, then we have $t_{i-1}^{\boldsymbol{x}} \ge d_{i-1}^{\boldsymbol{x}} = d_i^{\boldsymbol{u}} = t_i^{\boldsymbol{u}} > t_{i+k}^{\boldsymbol{z}} \ge t_{i+k-1}^{\boldsymbol{y}}$. This clearly contradicts the fact that $\boldsymbol{x} \prec \boldsymbol{y}$.

This implies that $t_i^{\boldsymbol{u}} = t_{i-1}^{\boldsymbol{u}} + 2$.

If $t_{i+k}^{\boldsymbol{z}} \geq t_{i+k-1}^{\boldsymbol{z}} + 2$, we have $t_i^{\boldsymbol{u}} = t_{i-1}^{\boldsymbol{u}} + 2 > t_{i+k}^{\boldsymbol{z}} \geq t_{i+k-1}^{\boldsymbol{z}} + 2$. Therefore, $t_{i-1}^{\boldsymbol{u}} > t_{i+k-1}^{\boldsymbol{z}}$. Iteratively substitute $i = i-1$ until either $i = m$ in state $\boldsymbol{u}$ or $i+k < n$ in state $\boldsymbol{z}$. If $i = m$, then $t_{m-1}^{\boldsymbol{u}} > t_{i+k-1}^{\boldsymbol{z}}$ which contradicts either Case 2 or Case 3 depending on whether $i+k < n$ or $i+k \geq n$, respectively. If $i+k < n$, then $t_{i-1}^{\boldsymbol{u}} > t_{i+k-1}^{\boldsymbol{z}}$ contradicts Case 2.

Hence, (10) holds for Case 4.

Since the four cases exhaustively include all possibilities, the lemma follows. ■

We now proceed to the proof of Theorem 1.

Proof of Theorem 1:

Proof: For $t = 0$, we have $\boldsymbol{X}^0(0) = \boldsymbol{X}(0)$, and hence $\boldsymbol{X}^0(t) \prec \boldsymbol{X}(t)$ at $t = 0$. Assume that $\boldsymbol{X}^0(t) \prec \boldsymbol{X}(t)$ is true for some t. We show that it holds for $t+1$ as well. Let $\boldsymbol{I}(t+1)$ be the activation vector under some policy π at $t+1$. Then from Lemma 3 we have

$$(\boldsymbol{X}^0(t) + \boldsymbol{R}g_A(\boldsymbol{X}^0(t))) \prec \boldsymbol{X}(t) + \boldsymbol{RI}(t+1). \qquad (11)$$

The arrival vector $\boldsymbol{A}(t+1)$ for the equivalent linear network can be written as

$$\boldsymbol{A}(t+1) = \sum_{i=1}^{N} A_i(t+1)\boldsymbol{e}_i. \qquad (12)$$

Hence from Lemma 4 and the relation (11) we can see that

$$\begin{aligned} \boldsymbol{X}^0(t+1) \qquad &= \boldsymbol{X}^0(t) + \boldsymbol{R}g_A(\boldsymbol{X}^0(t)) + \sum_{i=1}^{N} A_i(t+1)\boldsymbol{e}_i \\ &\prec \boldsymbol{X}(t) + \boldsymbol{RI}(t+1) + \sum_{i=1}^{N} A_i(t+1)\boldsymbol{e}_i \\ &= \boldsymbol{X}(t+1). \end{aligned}$$

Since we are only interested in the sum of all the queue lengths, the result for the equivalent linear network holds for the forest topology as well. ■

Thus, this completes our analysis for Class $\mathcal{A}$ forests, and we understand that a sample-path optimal policy for these forests is to simply construct an equivalent linear network, and schedule packets in the equivalent linear network according to the one-hop interference model.

B. Class $\mathcal{B}$

We now analyze Class $\mathcal{B}$ forests. We recall that these are the class of single-hop forests where all the roots of the forest have exactly one common child, and at most one root has other children.

We define the following notation for this class. We denote the root of the forest that has multiple children as S, and the set of other roots (that have one common child) as S'. We denote the child node that is connected to multiple roots as M, and the set of children whose only parent is node S as M'.

A node in M' can simultaneously transmit to S, when M transmits to a node in S'. However, when M transmits to S, no other links in the network can be activated. For example, in Figure 1(b), the root S_1 represents S, the roots S_2 and S_3 belong to the set S', the child D represents M, and the children E, F, and G belong to the set M'. When D transmits to S_2 or S_3, one of the children E, F, or G can simultaneously transmit to S_1 under the one-hop interference model. However, when D transmits to S_1, no other transmissions can be scheduled in the network.

We now develop a causal sample-path optimal policy π_B for Class $\mathcal{B}$ forests.

Policy π_B: Policy π_B for Class $\mathcal{B}$ forests uses the following schedule during any given slot.

- If there is at least one packet in any node in M' (destined to S), and at least one packet in M destined to one of the roots in S', then schedule a packet from one such node (having a packet) in M', and a packet from M to one such root (for which a packet is destined) in S' simultaneously.
- If there is at least one packet in any node in M' (destined to S), no packets in M destined to any of the roots in S', and at least one packet in M destined to S, then schedule a packet from M to S.
- If there is at least one packet in any node in M' (destined to S), and no packets in M, then schedule a packet from one such node (having a packet) in M'.
- If there are no packets in any node in M', and at least one packet in M destined to S, then schedule a packet from M to S.
- If there are no packets in any node in M', no packets in M destined to S, and at least one packet in M destined to one of the roots in S', then schedule a packet from M to one such root in S' (for which a packet is destined).

The intuition behind policy π_B is the following. During any slot, if it is possible to schedule two packets simultaneously, then π_B will schedule two packets. Since no more than two packets can leave the system during any slot, the sum of the queue lengths is minimized. If only one packet can be scheduled, then it either means that there are no packets in any node in M' destined to S, or that there are no packets in M destined to any root in S'. In this case, π_B prioritizes packets from M to S. The reason is that when a packet is scheduled from M to S, no other packet can be scheduled simultaneously. On the other hand, if a future arrival occurs such that there is a packet from one of the nodes in M' destined to S, and a packet in M destined to one of the roots in S', then these two packets can simultaneously leave the system in this slot. For instance, in Figure 1(b), suppose that there is one packet at D destined to S_1, and one packet at E destined to S_1. We schedule the packet from D to S_1 so that in the next slot if a packet arrives at D destined to S_2 (say), two packets (E to S_1 and D to S_2) can simultaneously leave the system in this slot. On the other hand, if we had scheduled the packet from E to S_1, then for the same arrival pattern, it would have taken two additional slots for the packets at D to S_1 and S_2 to leave the system.

We now prove that π_B is sample-path optimal for Class $\mathcal{B}$ forests.

Theorem 2. *Policy π_B is a causal sample-path optimal scheduling policy for Class $\mathcal{B}$ forests, i.e., when policy π_B is used to schedule any given Class $\mathcal{B}$ forest, at each time* *slot, for any sample-path traffic arrival pattern, the sum of the queue lengths of all the nodes in the given Class $\mathcal{B}$ forest is minimum among all policies.*

Proof: First, it is clear that policy π_B is causal. To show that it is sample-path optimal, we prove that the number of packets leaving the system is maximum at any given time slot. We do this by induction.

Consider time slot $t = 1$. The maximum number of packets that can leave the system is two during any time slot (if there exists a packet from one of the nodes in M' to S, and a packet from M to one of the roots in S'). If there exists two such packets at slot $t = 1$, policy π_B will schedule these packets. Otherwise, at most one packet can leave the system during the slot, and if there exists a packet in the system, π_B will schedule it. Hence, for time slot $t = 1$, the number of packets leaving the system is maximum. Hence, the sum of the queue lengths at the end of the first slot is minimum among all policies.

Assume that the result is true at time slot t. Let the number of packets that left the system by slot t be m, and this is maximum.

Consider time slot $t + 1$.

If there are no packets in the system, then the result is obvious.

If there exists a packet from one of the nodes in M' to S, and a packet from M to one of the roots in S', then π_B will schedule two packets, and the number of packets that leave the system after $t + 1$ slots is $m + 2$, which is maximum by the induction hypothesis, and by the fact that no more than two packets can leave the system in any single time slot.

Consider the case in which either none of the nodes in M' have a packet to S, or M does not have a packet to any of the roots in S'. In this case, at most one packet can leave the system. We now show that no more than $m + 1$ packets can leave the system after $t + 1$ slots according to any scheduling policy. We consider the following cases.

- Suppose that M has no packet to any of the roots in S', and at least one of the nodes in M' have at least one packet to S. Then, π_B will schedule exactly one packet. This packet would either be from M to S if M has such a packet, or it would be from one of the nodes in M' to S (if M has no packet destined to S). In either case, $m + 1$ packets would leave the system after $t + 1$ slots. Now, suppose that according to some policy π, two packets leave the system in slot $t + 1$. This means that, according to π, M has at least one packet to one of the roots in S', and at least one of the nodes in M' have a packet to S. Policy π_B (in a previous slot) would have scheduled a packet from M to one of the roots in S' either if one of the nodes in M' had a packet to S, or if there were no packets destined to S. Since a packet from M to one of the roots in S' exists according to policy π, then this packet was either not scheduled along with a packet from one of the nodes in M' to S, or no packet was scheduled during this slot. In either scenario, the number of packets that could have exited the system by slot t can at most be $m - 1$. Therefore, at most $m + 1$ packets can exit the system according to any scheduling policy after $t+1$ slots.
- Suppose that M has at least one packet to at least one of the roots in S', and none of the nodes in M' have any packets. Again, π_B will schedule exactly one packet, and this packet would either be from M to S if M has such a packet, or it would be from M to one of the roots in S' (if M has no packet destined to S). The argument that no more than $m + 1$ packets can leave the system after $t + 1$ slots is now similar to the previous case.

Hence, the number of packets leaving the system after $t+1$ slots is also maximum.

Hence, policy π_B is a causal sample-path optimal policy for Class $\mathcal{B}$ forests. ■

This completes our analysis for Class $\mathcal{B}$ forests. We will see in Section IV that for any other single-hop forest structure, there does not exist a causal sample-path optimal policy. This means that no two roots can have more than one child, and no two children can have more than one root as their parents.

C. Class $\mathcal{C}$

We finally investigate Class $\mathcal{C}$ forests. These forests are actually trees as they have only one root. We consider trees where all but one of the root's children is not a leaf node. The structure of the sub-tree rooted at the child (of the root) that is not a leaf node can be arbitrary. We also note that if the root has only one child, and that child is not a leaf node, then this structure also belongs to Class $\mathcal{A}$. In fact, we use this information to develop a scheduling policy.

We propose a causal policy π_C for this class, and show that it is sample-path optimal.

Policy π_C: This policy uses the following scheduling rules at any time slot.

- If the root's child that is not a leaf node has a packet, then schedule the root's child. Do not schedule the other children of the root. Schedule the rest of the tree according to policy π_A. If all of the root's children are leaf nodes, pick any one of them that has a packet to transmit, and schedule it. Do not schedule the other children.
- If the root's child that is not a leaf node does not have a packet, schedule any one of the root's other children that has a packet, and do not schedule the other children. Schedule the rest of the tree according to policy π_A.

Theorem 3. *Policy π_C is a causal sample-path optimal scheduling policy for trees where the root has multiple children, and all but one of the children are leaf nodes.*

Proof: Clearly, policy π_C is causal. To show that it is sample-path optimal, we prove that the number of packets leaving the system is maximum at any given time slot. We do this by induction. If the number of packets that have exited the system by any slot is maximum among all policies, it implies that the sum of the queue lengths of all the nodes in the network is minimum at any time slot.

Consider time slot $t = 1$. If one of the root's children have a packet, then this packet will be scheduled, and one packet will exit the system. Clearly, this is maximum among all policies since by the one-hop interference model no more than one packet can leave the system in any time slot for Class $\mathcal{C}$ forests. If none of the root's children have a packet, then no packet will reach the root, and this is also maximum among all policies.

Assume that the result is true for time slot t. Suppose that the number of packets that have exited the system by t is m according to policy π_C, i.e., m is the maximum number of packets that could have exited the system according to any policy after t slots.

Consider slot $t+1$. There are two cases.

Case 1: At least one of the root's children have a packet to schedule.

In this case, according to π_C, exactly one of these children will be scheduled, and the number of packets that exit the system by $t+1$ is $m+1$. This is maximum since at most one packet can exit during a time slot, and the maximum number of packets that have exited the system by t is m.

Case 2: None of the root's children have a packet to schedule.

This means that all the packets from all the root's children that are leaf nodes have exited the system, and there are no new arrivals at these nodes. Further, the branch corresponding to the root's child that is not a leaf node is scheduled according to policy π_A at all time slots. Therefore, the number of packets that have exited from this branch at any time slot is maximum because of the optimality of π_A for Class $\mathcal{A}$ forests. Hence, the maximum number of packets that have exited the system by $t+1$ is also m.

Therefore, by induction, at each time slot, for any arrival pattern, the number of packets that have exited the system is maximum among all policies, and hence π_C is a causal sample-path optimal scheduling policy for Class $\mathcal{C}$ forests. ■

IV. Non-Existence of Sample-Path Optimal Policies

In the previous section, we studied three classes of forests, Classes $\mathcal{A}$, $\mathcal{B}$, and $\mathcal{C}$, for which there exists causal sample-path optimal policies under the one-hop interference model. In this section, we show that for any other forest structure, there is a traffic arrival pattern for which there cannot exist a causal sample-path optimal policy.

We start by considering a tree structure that is not a Class $\mathcal{C}$ tree, i.e., the root has more than one non-leaf child.

Theorem 4. *There exists no causal sample-path optimal scheduling policy for a tree structure in which the root has two or more non-leaf children, i.e., without knowing future traffic arrivals, for such trees, there is at least one type of traffic arrival pattern for which no sample-path optimal scheduling policy exists.*

Proof: Let us begin by first considering the smallest tree with multiple non-leaf children, as shown in Figure 3. Assume that at $t=0$, nodes A and B have one packet each while C and D have no packets. In the absence of information of future traffic arrivals, we have no choice but to pick one of A or B to schedule at $t=0$.

Suppose we pick A to schedule. So at $t=1$, A has no packets left while B has one packet left. Now suppose that

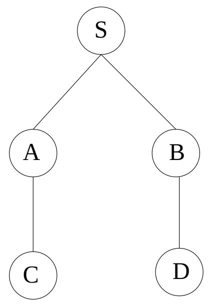


Fig. 3. Tree with no sample-path optimal scheduling policy

at $t=1$, we have an arrival at D. Assume that there are no arrivals at any other node, and that there are no future arrivals in the system. It then takes an additional three time slots for the packets at B and D to exit the system.

Suppose that we had picked B to schedule at $t=0$. Then, at $t=1$, A has one packet left to schedule while B has no packets left. In this case, at $t=1$, A would have transmitted to the root and D would have transmitted its packet to B simultaneously. At $t=2$, B would have transmitted this packet to the root. Therefore, it just takes two time slots for all the packets to exit the system.

Note that if we had picked B to schedule at $t=0$ without knowing about future arrivals, an arrival at C at $t=1$ would have ensured that picking B in the earlier time slot was sub-optimal.

Thus, this example shows that even for this simple tree with four nodes, there exists no causal sample-path optimal policy. It is now straightforward to see that for a general tree where the root has multiple children that are not leaf nodes, there exists no causal sample-path optimal policy. This is because such a tree would contain the above example as a part of the structure. So by simply considering the arrival pattern described above, and assuming that there are no packets and arrivals in the rest of the tree, the same argument would apply. ■

We can now completely classify tree structures for which causal sample-path optimal policies exist under the one-hop interference model.

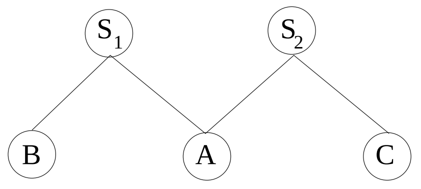
Theorem 5. *A causal sample-path optimal scheduling policy under the one-hop interference model exists in a given tree structure if and only if at most one of the root's children is not a leaf node.*

Proof: The result follows from Theorem 3 for Class $\mathcal{C}$ forests, and Theorem 4. ■

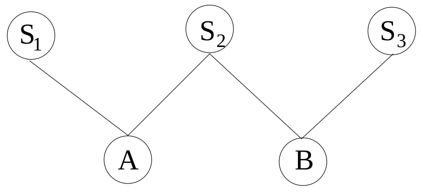
We now consider a forest in which more than one root has more than one child.

Theorem 6. *There exists no causal sample-path optimal scheduling policy in a forest in which there exists at least two roots each with at least two children.*

Proof: First, note that we only consider connected forests. Therefore, the simplest structure in which two roots have two children each is given in Figure 4(a). S_1 and S_2 are the roots. A is the common child of S_1 and S_2, B is a child of S_1, and C is a child of S_2.



(a) Two roots each having two children



(b) Two children each have two roots as parents

Fig. 4. Modified (single-hop) Class B forests

Consider the following traffic arrival pattern. At time slot $t = 0$, there exists two packets at A, one destined to S_1, and the other to S_2. There are no other packets in the system. Without knowing about future arrivals, we can either schedule the packet destined to S_1, or the packet destined to S_2 during the first slot.

Suppose that we schedule the packet destined to S_1. Suppose that at slot $t = 1$, a packet arrives at node C (destined to S_2). Then, it clearly takes two more slots for the packet from A to S_2, and C to S_2 to leave the system. On the other hand, if we had scheduled the packet from A to S_2 during the first slot, then during the second slot, the packet from A to S_1, and the packet from C to S_2 can be simultaneously scheduled. Therefore, it takes only one more time slot for the packets to leave the system in this scenario.

Suppose that we schedule the packet destined to S_2 during the first slot. Then, at slot $t = 1$, if a packet had arrived at B (destined to S_1), then it would have taken two more slots for the packet from A to S_1, and B to S_1 to leave the system. However, if we had scheduled the packet from A to S_1 during the first slot, it would have only taken one time slot for the remaining packets to leave the system.

Thus, without knowing whether a packet arrival is going to occur at B or C, there doesn't exist a sample-path optimal policy.

As argued previously, since any forest structure in which there exists at least two roots each with at least two children contains the forest structure in Figure 4(a), by simply considering the traffic arrival pattern described above, and no packets in the rest of the system, we can conclude that there exists no causal sample-path optimal policy in such forests. ■

We next consider the structures where at least two children have at least two roots as their parents.

Theorem 7. *For forest structures in which at least two children have at least two roots as their parents, there exists no causal sample-path optimal scheduling policy.*

Proof: As before, we look at the simplest forest structure in which two children have two roots as their parents. Consider the forest structure in Figure 4(b) in which there are three roots S_1, S_2, and S_3. Node A has S_1 and S_2 as its parents, and node B has S_2 and S_3 as its parents.

Consider the following traffic arrival pattern. At time slot $t = 0$, there exists one packet at A destined to S_2, and one packet at B also destined to S_2. We can either decide to schedule A or B during the first slot. If we decide to schedule A, and at time slot $t = 1$, we get a packet at B destined to S_3, then it would take two additional slots for the two packets at B to leave the system. On the other hand, if we had scheduled B during the first slot, then during the second slot, the packet from A to S_2, and the packet from B to S_3 could have been scheduled simultaneously (since these links do not interfere under the one-hop interference model), and would hence only require one more time slot to leave the system.

The argument is similar if we had decided to schedule B during the first slot.

Thus, without knowing the future traffic arrival pattern, there exists no sample-path optimal scheduling policy for this structure under the one-hop interference model. ■

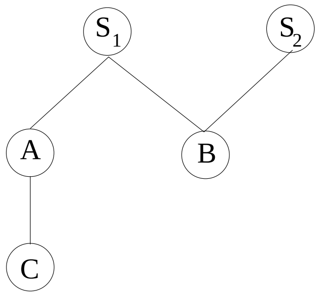
From Theorems 6 and 7, we can understand the reasoning behind the topology restriction for Class $\mathcal{B}$ forests. Clearly, if no two roots can each have at least two children, and no two children can each have at least two roots as parents, then single-hop forest structures can have only one root having multiple children, and only one child having multiple parents.

We now look at multi-hop forest structures. We first consider the structure in which one of the children that has only one root as its parent is not a leaf node.

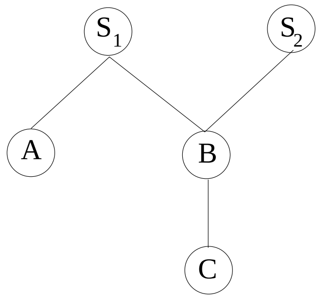
Theorem 8. *Consider a modified Class $\mathcal{B}$ forest structure in which one of the children that has only one root as its parent is not a leaf node. There exists no causal sample-path optimal scheduling policy for such forest structures.*

Proof: First, if the root in the Class $\mathcal{B}$ forest having multiple children has more than one child that is not a leaf node, then we know that there can exist no sample-path optimal policy by Theorem 4.

Consider the simplest structure (Figure 5(a)) in which we have two roots, S_1 and S_2. S_1 has A and B as its children, and S_2 has B as its only child. Also, node C is the child of node A. Clearly, without node C, this structure is a Class $\mathcal{B}$ structure.



(a) Modified Class $\mathcal{B}$ forest with node A having a child



(b) Modified Class $\mathcal{B}$ forest with node B having a child

Fig. 5. Modified (multi-hop) Class $\mathcal{B}$ forests

Consider the following traffic arrival pattern. Suppose that at time slot $t = 0$, A has one packet destined to S_1, and B

has one packet also destined to S_1. There are no other packets in the system. Without knowing about future arrivals, we can either schedule A or B during the first slot. As argued for earlier proofs, we show that in either case, there is a traffic arrival pattern such that a sample-path optimal policy cannot exist.

First, if we were to schedule A during the first slot, and if a packet destined to S_2 arrives at B at slot $t = 1$, then it would take two more slots for the packets at B destined to S_1 and S_2 to leave the system. On the other hand, if we had scheduled B during the first slot, then it would have taken only one additional slot since the packet from A to S_1 and the packet from B to S_2 could have been simultaneously scheduled.

However, if we were to always schedule B during the first slot, and if a packet destined to S_1 arrives at node C at slot $t = 1$, then it would take three more slots for the packets at A and C to reach S_1. On the other hand, if we had scheduled A during the first slot, then it is easy to see that it would only have taken two additional slots for these packets to leave the system.

Thus, there exists no causal sample-path optimal scheduling policy for such forests. ■

We now consider structures in which the child that is common to all the roots is not a leaf node while the other children having the same root as their parent are leaf nodes.

Theorem 9. *Consider a modified Class $\mathcal{B}$ structure in which the child that is common to all the roots is not a leaf node while the other children having the same root as their parent are leaf nodes. There exists no causal sample-path optimal scheduling policy for such forest structures.*

Proof: We first recall that if more than one child is not a leaf node, then there exists no causal sample-path optimal scheduling policy for such a structure by Theorem 4.

Consider the simplest structure (Figure 5(b)) in which there are two roots, S_1 and S_2. S_1 has A and B as its children, and S_2 has B as its only child. Further, node C is a child of node B.

Consider the following traffic arrival pattern. Suppose that node C has one packet destined to S_1 and one packet destined to S_2 at time slot $t = 0$. Suppose that we schedule the packet destined to S_1, and suppose that at time slot $t = 1$, a packet arrives at node A destined to S_1. Then, by slot $t = 2$, only one packet can leave the system. However, if we had scheduled the packet destined from C to S_2 during the first slot, then it is easy to see that two packets could have left the system by slot $t = 2$.

Now, suppose that the policy was to always schedule the packet destined to S_2 from C. Then, at slot $t = 1$, we will have one packet at B destined to S_2, and one at C destined to S_1. Suppose that there are no new arrivals at slot $t = 1$. Then, we will schedule the packet from B to S_2 during the second slot. Again, suppose that there are no new arrivals at slot $t = 2$. We will now schedule the packet from C during the third slot. Now, suppose that a packet arrives at A destined to S_1. Then, it will take two more slots for the packets at A and B to reach S_1. On the other hand, if we had scheduled the packet from C destined to S_1 during the first slot, it can be easily shown that it would have only taken one additional slot for these packets to leave the system.

Therefore, no causal sample-path optimal policy exists for these forests. ■

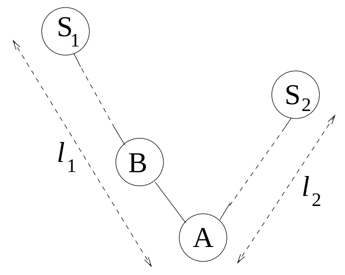
To complete our analysis on multi-hop forests, we now study structures in which two branches with a common node lead to two different roots. This common node must be at a depth greater than one from at least one of the roots. If the common node is at depth one from both the roots, then this structure belongs to one of the single-hop structures discussed previously.

Theorem 10. *Consider a forest with at least two roots. If there exists two roots in this forest such that the common node that branches out to these roots is l_1 hops away from one root, and l_2 hops away from the other, and either $l_1 > 1$, or $l_2 > 1$, there exists no causal sample-path optimal policy in this structure.*

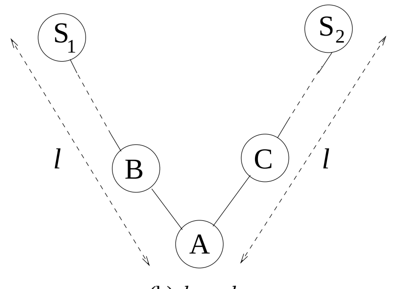
Proof: We consider two cases.

Case 1: $l_1 \neq l_2$.

Suppose that node A (Figure 6(a)) is the common node that branches out to the roots S_1 and S_2. Let A be l_1 hops away from S_1, and l_2 hops away from S_2. WLOG, assume that $l_1 \neq 1$, and $l_1 > l_2$. Since $l_1 > 1$, let B be the parent of A in the branch leading to the root S_1. Consider the following traffic arrival pattern. Suppose that at time slot $t = 0$, there exists one packet at A destined to S_1, and another packet at A destined to S_2.



(a) $l_1 \neq l_2$



(b) $l_1 = l_2$

Fig. 6. $l_1 > 1$ or $l_2 > 1$

Since $l_2 < l_1$, we have to schedule the packet from A to S_2 during the first slot. Otherwise, a policy that schedules this packet will have one packet less in the system at time $t = l_2$, while any other policy will have both the packets in the system at slot $t = l_2$. Suppose that we schedule the packet from A to S_2 during the first slot. It would then take l_1 additional time slots for all the packets to leave the system. This can be reasoned as follows. The packet destined to S_2 will not interfere with the packet destined to S_1 in any of the future slots. So, it will leave the system in $l_2 - 1$ slots. Also, it will take l_1 slots for the other packet from A to reach S_1.

On the other hand, suppose that we had scheduled the packet from A to S_1 during the first slot. In this case, at slot $t = 1$, we will have one packet at node B destined to S_1, and one packet at A destined to S_2. In the second slot, the packet at B, and the packet at node A destined to S_2 will be scheduled simultaneously. From this slot, the packet destined to S_2 will reach S_2 in $l_2 - 1$ slots, and the packet at B's parent destined to S_1 will reach S_1 in $l_1 - 2$ slots. Hence, from the first slot, it only takes $l_1 - 1$ additional slots for all the packets to leave the system.

Since we have to schedule the packet from A to S_2 during the first slot irrespective of whether there exists a packet at A destined to S_1, it follows that *even in the absence of future arrivals, there does not exist any optimal scheduling policy that minimizes the sum of the queue lengths of all the nodes in the system at each time slot for this forest structure*.

Case 2: $l_1 = l_2 = l > 1$.

Consider Figure 6(b) in which node A is the common node that branches out to the roots S_1 and S_2. Node A is l hops away from both S_1 and S_2. Let B be the parent of A in the branch leading to S_1, and C be the parent of A in the branch leading to S_2. Suppose that at time slot $t = 0$, we have two packets at A, one destined to S_1, and the other destined to S_2.

Suppose that we schedule the packet destined to S_1 during the first slot. If we do this, and a packet destined to S_2 arrives at A at slot $t = 1$, it would take $l+2$ additional slots for all the packets to leave the system (as argued in Case 1). On the other hand, if we had scheduled the packet destined to S_2 during the first slot, then at slot $t = 1$, we will have one packet at A destined to S_2, one packet at C destined to S_2, and one packet at A destined to S_1. In the second slot, we would have scheduled the packet at C destined to S_2, and the packet at A destined to S_1 simultaneously. It would have then taken an additional l slots for all the three packets to leave the system (since none of the packets would have interfered with the others during any of the future slots, and the farthest packet is l hops from S_2). Therefore, it would have only taken $l + 1$ additional slots from the first slot for all the packets to leave the system.

Suppose that we schedule the packet destined to S_2 during the first slot. If we do this, and a packet destined to S_1 arrives at A at slot $t = 1$, it would again take $l + 2$ additional slots for all the packets to leave the system. However, if we had scheduled the packet destined to S_1 during the first slot, then, as argued above, it would have only taken $l + 1$ additional slots for all the packets to leave the system.

Thus, there exists no causal sample-path optimal scheduling policy for such forests. ∎

Theorem 11. *There exists a causal sample-path optimal scheduling policy for a given forest structure under the one-hop interference model if and only if the structure belongs to Classes $\mathcal{A}$, $\mathcal{B}$, or $\mathcal{C}$.*

Proof: The result for the existence of causal sample-path optimal policies for Classes $\mathcal{A}$, $\mathcal{B}$, and $\mathcal{C}$ follows from Theorems 1, 2, and 3, and the result for the non-existence of causal sample-path optimal policies for any other structure follows from Theorems 4, 6, 7, 8, 9, and 10. ∎

We have thus completely characterized the existence of causal sample-path optimal scheduling policies in any forest structure under the one-hop interference model. We again note that we have only considered connected forests, and that if we had disconnected structures, we simply have to apply Theorem 11 separately to each connected structure. If there exists causal sample-path optimal policies in all the connected structures, there exists a causal sample-path optimal policy for the entire (disconnected) structure. If there does not exist a causal sample-path optimal policy for even one of the connected structures, there does not exist a causal sample-path optimal policy for the entire (disconnected) structure.

V. Discussion

While having a sample-path optimal policy is ideal, they exist for very limited topologies. We briefly discuss this limitation, and also make a number of interesting observations about other metrics for delay studied in the literature for tree structures. Designing such policies for forests is an open problem.

- *Non-existence of sample-path optimal policies*: As seen from the previous sections, sample-path optimal policies exist in very limited forest topologies. Since this metric requires optimality at each time slot and any traffic arrival pattern, it is unlikely to exist for many topologies. As we have shown, simple traffic arrival patterns can be constructed to prove that such policies do not exist for many topologies. The primary reason for its existence in the three classes of forest topologies identified here is the simplicity of the topology for Class $\mathcal{B}$ forests, and the relationship to scheduling in an equivalent linear network for Classes $\mathcal{A}$ and $\mathcal{C}$.
- *Large deviations metric*: In [11], Venkataramanan et al., have shown that for the convergecasting problem in general trees, Tassiulas's policy is optimal in the large deviations sense. However, this policy is actually not even evacuation time optimal, i.e., even in the absence of arrivals, it does not minimize the time by which all the packets leave the system. For instance, consider the following tree (Figure 7) with four nodes A, B, C, D, and a root. Suppose that A, B, and D have one packet each. According to the policy in [11], either A or B can be chosen to schedule arbitrarily during the first time slot. However, one can easily see that if A were chosen to schedule during the first time slot, the time to evacuate the system is 3 slots. But if B were chosen to schedule during the first time slot, the time to evacuate the system is 4 slots.
- *Evacuation time optimality*: In [2], Florens et al., have proposed an evacuation time optimal policy for general trees by prioritizing the branches of the tree according to the time required to evacuate each individual branch. From the proof of Theorem 10, we can see that a policy that is evacuation time optimal need not minimize the sum of the queue lengths of all the nodes at each time slot (even if it evacuates the system in minimum time).

Both these metrics are interesting since they provide optimal scheduling policies (in their respective senses) for general trees. However, the evacuation time metric does not consider arrivals, and there exists instances where the policy based on large deviations is not evacuation time optimal. Therefore, care should be taken when designing scheduling algorithms based on these policies.

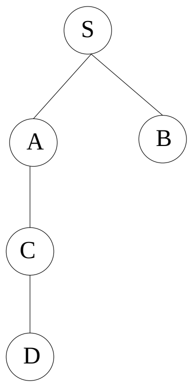


Fig. 7. Arbitrarily choosing branches is not evacuation time optimal

VI. Conclusion

We studied sample-path optimal scheduling for forest structures under the one-hop interference model. We showed that sample-path optimal policies exist in three classes of forests. We observed an interesting relationship showing that some of these forests can be scheduled optimally according to a schedule in an equivalent linear network. Further, we showed that these are the only forest structures for which there exists sample-path optimal scheduling policies under the one-hop interference model. The fact that causal sample-path optimal policies exist in very limited cases is, however, a limitation of this metric. This emphasizes the need to study other "softer" metrics for delay. For instance, when the stochastic nature of the arrival pattern is known, one could study the expected delay. Further, while we only focused on the one-hop interference model, investigating this problem for other interference models is also of importance. These are challenging problems for future work.

References

[1] J. C. Bermond, L. Gargano, and A. A. Rescigno. Gathering with minimum completion time in sensor networks. *Journal of Interconnection Networks*, 11(1-2), 2010.

[2] C. Florens, M. Franceschetti, and R. J. McEliece. Lower bounds on data collection time in sensory networks. *IEEE Journal on Selected Areas in Communications*, 22(6), 2004.

[3] L. Gargano and A. A. Rescigno. Collision-free path coloring with application to minimum-delay gathering in sensor networks. *Discrete Applied Mathematics*, 157(8), 2009.

[4] G. R. Gupta and N. B. Shroff. Delay analysis and optimality of scheduling policies for multi-hop wireless networks. *IEEE/ACM Transactions on Networking*, 19(1), 2011.

[5] S. Hariharan and N. B. Shroff. Deadline constrained scheduling for data aggregation in unreliable sensor networks. In *Proceedings of the 9^{th} International Symposium on Modeling and Optimization in Mobile, Ad Hoc, and Wireless Networks (WiOpt)*, 2011.

[6] S. Hariharan and N. B. Shroff. On optimal dynamic scheduling for sum-queue minimization in trees. In *Proceedings of the 9^{th} International Symposium on Modeling and Optimization in Mobile, Ad Hoc, and Wireless Networks (WiOpt)*, 2011.

[7] S. Hariharan and N. B. Shroff. On optimal energy efficient convergecasting in unreliable sensor networks with applications to target tracking. In *Proceedings of the Twelfth ACM International Symposium on Mobile Ad Hoc Networking and Computing (MobiHoc)*, 2011.

[8] S. Hariharan and N. B. Shroff. On sample-path optimal dynamic scheduling for sum-queue minimization in trees under the k-hop interference model. In *Proceedings of IEEE INFOCOM*, 2012.

[9] T. Ji, E. Athanasopoulou, and R. Srikant. On optimal scheduling algorithms for small generalized switches. *IEEE/ACM Transactions on Networking*, 18(5), 2010.

[10] L. Tassiulas and A. Ephremides. Dynamic scheduling for minimum delay in tandem and parallel constrained queue networks. *Annals of Operations Research*, 48, 1994.

[11] V. J. Venkataramanan and X. Lin. Low-complexity scheduling algorithm for sum-queue minimization in wireless convergecast. In *Proceedings of IEEE INFOCOM*, 2011.

Srikanth Hariharan received his Ph.D. in Electrical and Computer Engineering from the Ohio State University in 2011, M.S. in Electrical and Computer Engineering from Purdue University in 2007, and B.Tech. in Electrical Engineering from the Indian Institute of Technology Madras in 2006. He is currently with the Analysis and Optimization department in AT&T Labs. His research interests include analysis and optimization of LTE/UMTS networks, data aggregation in wireless sensor networks, multi-hop wireless scheduling, and wireless security.

Ness B. Shroff (F'07) received his Ph.D. degree from Columbia University, NY, in 1994 and joined Purdue University as an Assistant Professor. At Purdue, he became Professor of the School of Electrical and Computer Engineering in 2003 and director of CWSA in 2004, a university-wide center on wireless systems and applications. In July 2007, he joined The Ohio State University as the Ohio Eminent Scholar of Networking and Communications, a chaired Professor of ECE and CSE. He is also a guest chaired professor of Wireless Communications and Networking in the department of Electronic Engineering at Tsinghua University. His research interests span the areas of wireless and wireline communication networks. He is especially interested in fundamental problems in the design, performance, pricing, and security of these networks.

Dr. Shroff is a past editor for IEEE/ACM Trans. on Networking and the IEEE Communications Letters and current editor of the Computer Networks Journal. He has served as the technical program co-chair and general co-chair of several major conferences and workshops, such as the IEEE INFOCOM 2003, ACM Mobihoc 2008, IEEE CCW 1999, and WICON 2008. He was also a co-organizer of the NSF workshop on Fundamental Research in Networking (2003) and the NSF Workshop on the Future of Wireless Networks (2009). Dr. Shroff is a fellow of the IEEE. He received the IEEE INFOCOM 2008 best paper award, the IEEE INFOCOM 2006 best paper award, the IEEE IWQoS 2006 best student paper award, the 2005 best paper of the year award for the Journal of Communications and Networking, the 2003 best paper of the year award for Computer Networks, and the NSF CAREER award in 1996 (his INFOCOM 2005 paper was also selected as one of two runner-up papers for the best paper award).