Optimal Service Allocation For Multi-Hop Network With Intermittent Connectivity

Hussein Al-Zubaidy,
ECE, University of Toronto, Toronto, ON, Canada.
e-mail: hzubaidy@comm.utoronto.ca

***Abstract*—A network of queues is partitioned into K tandem, disjoint sets of parallel queues. A queue that belongs to set k is randomly connected to the queues in set $k+1$ with probability p. Time is slotted into fixed size transmission frames. A scheduler selects one packet per hop (set) per transmission frame to be forwarded to the next hop. Exogeneous arrivals may be inserted into set 1 queues only. This setup can be used to model a wireless relay network (WRN), where set 1 represents the subscriber stations (SS), while the other sets represent multi-hop relay stations (RS). In this paper, we investigate an optimal dynamic packet scheduling algorithm for this network. We define the optimal policy as the dynamic packet scheduling policy that minimizes, in a stochastic ordering sense, the process of cost function of the queue sizes in all hops. We consider a system with symmetrical connectivity and arrival distributions across the queues in each of the K sets. We prove that a policy that tries to balance the lengths of all queues in each of the K sets of queues in the system at every transmission frame is optimal. For the proof, we use coupling arguments. We also provide a low-complexity implementation algorithm and we prove that it will result in an optimal policy.**

I. Introduction

Wireless relays are intermediary wireless nodes that assist, with or without cooperation of other nodes in the network, with data transmission between a source and a destination. In a multi-hop wireless relay network (WRN), the data transmission may involve multiple wireless relays. The deployment of dedicated wireless relay nodes in fourth generation (4G) wireless networks design was recommended by the IEEE 802.16j task group [1] in order to enhance the capacity of these networks. Such configuration facilitates cooperation between relay nodes in order to achieve diversity gain. Cooperative communication was initially studied by [2], [3] and [4]. Among the benefits of such deployment are increased coverage and higher data rates. In addition, wireless relays typically have limited functionality and low power consumption, and are significantly cheaper than a base station. This makes them an attractive alternative for network expansion.

Most of the work in this area focuses on the utilization of diversity and multiplexing gain to improve performance criteria, such as capacity and bandwidth utilization, outage probability, error rate, etc. These are achieved using adaptive modulation techniques, distributed space-time coding, or error-correction coding.

In previous work [5], we looked at this problem from a different perspective, the dynamic packet scheduling perspective. The goal was to investigate an optimal packet scheduling policy on the uplink of a WRN. To do that, we presented a queuing model for a two-hop WRN. The wireless nodes were assumed to have random channel connectivity that was modelled as a random process. We used coupling arguments to prove the optimality of a class of policies, namely the class of most balancing (MB) policies. In this work, we extend our results to the general case of K-hop WRN (for $K \geq 2$). We develop a queueing model that captures packet buffering, scheduling and routing processes as well as the randomness of channel connectivity in multi-hop WRN. The model is then used to investigate the optimal dynamic packet scheduling policy in these networks. This extension is not a trivial one and the proofs of the presented results are much harder to obtain as we will show in Section IV.

Dynamic packet scheduling provides a mechanism for the redistribution of the available resources in order to improve network performance. Furthermore, it is possible to obtain packet scheduling policies that optimize performance criteria under various operating constraints. This motivated the work we present here. The stochastic nature of the wireless channel and the dynamic node configuration in wireless networks present a formidable challenge to our investigation. Therefore, it is often necessary to make suitable assumptions that result in mathematically tractable problem formulations. Otherwise, optimality results may not be attainable. Most of the assumptions used in this model are well vetted in the literature, e.g., [11], [12] and [13] .

In this work, we investigate an optimal dynamic packet scheduling policy for a K-hop wireless relay network (WRN). The network is composed of a base station (BS), a set of L_1 subscriber stations (SS) and $K-1$ tandem sets of relay stations (RS) where set $k, k = 2, 3, \ldots, K$ has L_k RS nodes, for any arbitrary $K < \infty$ and $L_k < \infty, k = 1, \ldots, K$. This network is modelled by K mutually exclusive[1] sets of infinite size queues (or stacks of queues, see Figure 1). Time is slotted into equal size transmission frames. The wireless channel connectivity is assumed to be a Bernoulli process. At any given time, a wireless link is assumed to be 'connected' with probability p and 'not connected' otherwise. We assume that the connectivity processes are independent across the wireless links. We further assume that relay nodes have the ability to schedule/route packets to the next hop. A transmission frame

[1]In most infrastructure based WRNs, the relays are uniformly distributed in order to increase the coverage area with minimum number of relay nodes. Therefore, this assumption is not far fetched. Furthermore, this assumption is essential to the optimality proof we present in this paper.

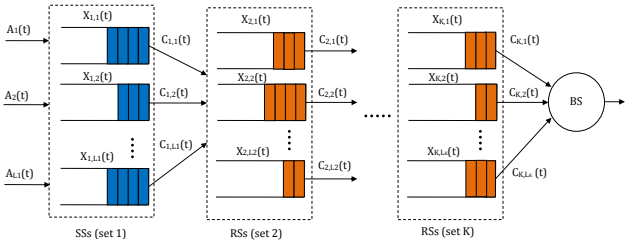

Fig. 1. A queueing model for a multi-hop wireless relay networks.

is divided into K slots; during the first slot a SS node is scheduled to transmit (to a selected RS on the second hop) and during the second slot, a RS node from the second set of nodes is scheduled to transmit to the next hop and so on. During the K^{th} slot, a RS node from the K^{th} set of nodes is scheduled to transmit to the base station.

In this work, we provide a proof using stochastic dominance [9] and coupling arguments [10], that a *most balancing* (MB) policy is optimal in that it minimizes, in stochastic ordering sense, the cost function under consideration[2]. The optimization cost function belongs to a set of monotone, non-decreasing functions of the process of queue occupancy in the system. The exact characterization of this class of functions is given in Section IV-B.

In the literature, similar problems can be found in the area of optimal control of queueing networks. Perhaps the earliest work related to ours is the one presented by Roseburg, Varaiya and Walrand in [8]. They investigated an optimal control policy of service rate in a system of two queues in tandem. They proved, using a dynamic programming argument, that a threshold policy is optimal in that it minimizes the expected cost function of the queue occupancy. Tassiulas and Ephremides [11] studied an optimal packet scheduling policy in a system of parallel queues with a single server and random connectivity; they showed that a LCQ policy, a policy that allocates the server to its longest connected queue, minimizes the total number of packets in the system. In [12] the authors studied a system of parallel queues with symmetrical statistics and K identical servers. At any given time slot: (i) a server is either connected (to all queues) or not connected, (ii) at most one server can be allocated to each scheduled queue. Using stochastic coupling arguments, the authors proved that allocating the K servers to the K longest connected queues at every time slot is optimal.

In [13], the authors studied the problem of optimal scheduling in a multi-server system of parallel queues with random queue-server connectivity. They relaxed assumptions (i) and (ii) in [12] above. They proved, using coupling arguments, that a Most Balancing (MB) policy is optimal in that it minimizes, in stochastic ordering sense, a cost function of the queues' occupancy.

In [15], the authors studied link scheduling in WRN with bandwidth and delay guarantees. They used simple directed graph to model the system. They proposed an efficient algorithm to provide delay guarantee over WRN. In [14], the authors proposed a cooperative multiplexing and scheduling algorithm for a WRN with a single relay node. They showed that this algorithm outperforms the traditional opportunistic scheduling in terms of spectral efficiency.

In WRN, the choices of relay node, relay strategy, and the allocation of power and bandwidth for each user are important design parameters. These parameters were investigated thoroughly in recent literature. Relay selection and cooperation strategies for relay networks have been investigated by [16] and [17] among others. Power control has been investigated by [18] and [19] and many others. However, the modern wireless networks are mostly IP-based. Therefore, dynamic packet scheduling optimization is an important problem in studying WRNs.

The main contributions of this work can be summarized by the following:

- We propose a queueing model to study the process of packet scheduling in a multi-hop wireless relay network, see Figure 1.
- We introduce (in Equation 12) the class of Most Balancing (MB) policies. We prove the optimality of MB packet scheduling policies (Theorem 2) in this model. The optimality was carried out in stochastic ordering sense. The cost that is minimized belongs to a set of functions of the SS and RS queue lengths.
- We suggest an algorithm to implement the MB policies in WRN. We provide a proof that this algorithm results in a MB policy.

The model we are presenting in this article differs from the previous work in that it contains K sets of parallel queues in tandem rather than just one or two. As we mentioned earlier, a transmission frame in our model is divided into K transmission slots, one per each hop. During transmission slot $k : k \in \{1, \ldots, K\}$ of each transmission frame t, the scheduler must decide which node, out of the L_k nodes in hop k, transmits a packet and which node in hop $k+1$ receives that packet. Solving this optimization problem is not straightforward. There are dependency issues between each two neighbouring sets of queues in the transmission chain that must be resolved before a rigorous proof can be carried out. This adds more complexity to the solution of this optimization problem. The approach we use in our proof (section IV) addresses this issue rigorously.

The rest of this article is organized as follows; in Section II, we provide description of the queueing model under investigation. In Section III, we introduce the class of "Most Balancing" policies. The optimality results are given in Section IV. Conclusions are given in Section V. Proofs for some of the results are given in the appendix.

II. Model Description and Problem Formulation

We use a discrete-time queueing system to model the WRN, as shown in Figure 1. The objective is to study the optimal dynamic packet scheduling policy for this model. We define the optimal policy as the one that minimizes, in stochastic ordering sense, a cost function of the queue lengths (to be defined shortly).

[2]Stochastic ordering is a stronger optimization notion than the expected cost minimization since the former implies the latter [7].

In this model, time is slotted into constant transmission intervals (time frames) denoted by t. Each frame is divided into K (transmission or scheduling) slots, each of which is good for transmitting one packet across a single hop, i.e., slot k is used to transmit one packet from a queue in set k to a queue in set $k+1$. During slot $k, k = 1, \ldots, K$, the following sequence of events occur: (a) the sizes and connectivities of queues that belong to the set k and their immediate neighbours in the set $k+1$ is observed, (b) the scheduler selects a control decision, then if $k = 1$ (c) the exogenous arrivals are added to their respective SS queues[3]. The scheduler decision during slot k involves (i) selecting a node in the set k to transmit to a relay node in the next hop (or to the BS if $k = K$); we denote this control by $U_{k,1}(t)$, (ii) selecting the relay node that the scheduled packet (in (i) above) is routed to; denote it by $U_{k,2}(t)$. The scheduler decision are sequentially executed with order[4] $\ldots, U_{K,1}(t-1); U_{1,1}(t), U_{1,2}(t), \ldots, U_{K,1}(t); U_{1,1}(t+1), \ldots$, where the semicolons indicate the limits for frame t. A packet that arrives during the current frame can only be considered for transmission in the subsequent frames. We assume that the scheduler has partial knowledge of the system states at every transmission (or scheduling) slot $k, k = 1, \ldots, K$. This knowledge includes the queue lengths and connectivities of all queues in the set $k : k \in \{1, \ldots, K\}$ and the set $k + 1$ for $k < K$.

A. Problem Formulation

Throughout this paper, we will use UPPER CASE, **bold face** and lower case letters to represent random variables, vector/matrix quantities and sample values respectively. In our notation, we define K dummy queues, one for each of the K sets of queues, that we denote by the index '0'. These queues are used to represent the idling action, i.e., a dummy packet is removed from queue 0 when no real packet from real queue is scheduled for transmission. We assume that the dummy queues have full connectivity at all times and are initially empty. The dummy queues are required in order to facilitate the mathematical formulation of this problem and to simplify notation. Let $\mathcal{L}^k = \{0, 1, \ldots, L_k\}, k = 1, \ldots, K$ be the set of indices for the k^{th} stack of queues in the system. For any transmission frame $t = 1, 2, \ldots$, we define the following:

1) $\mathbf{A}(t) = (A_0(t), A_1(t), \ldots, A_{L_1}(t))$, where $A_j(t)$ is the number of exogenous arrivals to SS queue j during transmission frame t.
2) $\mathbf{X}_k(t) = (X_{k,0}(t), X_{k,1}(t), \ldots, X_{k,L_k}(t)), k = 1, \ldots, K$ is the queue length vector for the nodes in set k (measured in number of packets) at the beginning of time frame t, where $X_{k,j}(t) \in \{0, 1, 2, \ldots\}$ and $X_{k,0}(1) = 0, \forall k$.
3) $\mathbf{C}_k(t) = (C_{k,0}(t), C_{k,1}(t), \ldots, C_{k,L_k}(t)), k = 1, \ldots, K$ is the channel connectivity for set k's nodes during slot k of transmission frame t, where $C_{k,0}(t) = 1, \forall k, t$.
4) $\mathbf{U}(t) = (U_{1,1}(t), U_{1,2}(t), \ldots, U_{K,1}(t))$, s.t. $U_{k,1}(t) \in \mathcal{L}^k, U_{k,2}(t) \in \mathcal{L}^{k+1}$, is the scheduler decision (or control), where $\mathbf{U}(t) = (i_1, j_1, i_2, j_2, \ldots, i_K)$ means that node i_k in set k is scheduled to transmit to node j_k in set $k+1$ during the k^{th} slot of frame t, $\forall k = 1, \ldots, K-1$, and node i_K is scheduled to transmit to base station during the K^{th} slot of transmission frame t.
5) We set $\mathbf{X}(t) = (\mathbf{X}_1(t); \ldots; \mathbf{X}_K(t))$ and $\mathbf{C}(t) = (\mathbf{C}_1(t); \ldots; \mathbf{C}_K(t))$. For ease of reference, we refer to $(\mathbf{X}(t), \mathbf{C}(t))$, as the system "state" (denoted by $\mathbf{S}(t)$) during transmission frame t.

To facilitate problem tractability we make several statistical assumptions for the model parameters. The exogenous arrival processes, to the SS nodes, $(A_i(t) \in \{0, 1\}, i = 1, \ldots, L_1)$ are assumed to be i.i.d. Bernoulli, with parameter q. The arrivals to any RS node in a subsequent set k during frame t is equal to the number of packets transmitted from a node in set $k-1$ to that RS node during slot $k-1$ of frame t. For convenience, we define $A_0(t) = W_{1,0}(t)$, where $W_{1,0}(t)$ is the number of packets withdrawn from queue 0 in set 1 during slot 1 of frame t, in order to ensure that $X_{1,0}(t) = 0$ for all t. Furthermore, transmitted dummy packets (i.e., fictitious packets from dummy queues) will not be added to the receiver queue (the RS queue that the packet is routed to). This assumption is needed to keep the book-keeping straight, since fictitious packet is generated only when there is no real packet transmission.

The connectivity processes $C_{k,i}(t)$ and $C_{k,j}(t)$, for all $k = 1, \ldots, K$, $i, j = 1, \ldots, L_k, i \neq j$ are assumed to be independent Bernoulli with connection probability p. We further assume that the arrival and connectivity processes are independent of each other.

To simplify problem formulation and the proofs, we define next the 'withdrawal' and the 'insertion' controls as a function of the scheduler control $\mathbf{U}(t)$.

B. Feasible Withdrawal/Insertion Control Vectors

At any given frame t, we define the *withdrawal vector* $\mathbf{W}_k(t)$ for set k as follows:

$$W_{k,i}(t) = \mathbb{1}_{\{U_{k,1}(t)=i\}}, \quad \forall i \in \mathcal{L}^k \tag{1}$$

where $\mathbb{1}_{\{B\}}$ denote the indicator function for condition B. $W_{k,i}(t)$ represents the number of packets withdrawn from queue i of set k during slot k of frame t. We also define the *insertion vector* for set $k \geq 2$ as:

$$V_{k,j}(t) = \mathbb{1}_{\{U_{k-1,2}(t)=j\}}, \quad \forall j \in \mathcal{L}^k \setminus \{0\} \tag{2}$$

and $V_{1,j}(t) = 0, \forall j$, i.e., there is no endogenous arrivals to queues in stack 1 (the SS nodes). $V_{k,j}(t)$ represents the number of packets inserted to RS queue j of set k during slot k of frame t. Note that we do not allow real packets to be inserted in the dummy queue. Similarly, we do not allow dummy packets to be inserted into real queues.

Given a (*feasible*) withdrawal control, the queue length process for SS nodes (stack 1) in the system described above evolves according to the following relation:

$$\mathbf{X}_1(t+1) = \mathbf{X}_1(t) - \mathbf{W}_1(t) + \mathbf{A}(t) \tag{3}$$

[3]The exogenous arrivals will be added at the end of slot 1 of each frame.

[4]Note that on the last hop (hop K) the only control a node scheduled for transmission has is to forward a packet to the base station.

Similarly, for any feasible withdrawal/insertion controls, the queue length process for RS nodes (stacks k=2,..., K) evolves according to the following relation:

$$\mathbf{X}_k(t+1) = \mathbf{X}_k(t) + \mathbf{V}_k(t) - \mathbf{W}_k(t) \tag{4}$$

Examining Equations (3) and (4) closely reveals that Equations (1) and (2) do not guarantee feasibility of the withdrawal/insertion vectors or the scheduling control $\mathbf{U}(t)$. To remedy that, we state the following feasibility condition:

$\mathbf{U}(t)$ is said to be a '*feasible scheduling control*' if the following condition is satisfied: '*a packet may only be withdrawn from a connected, non-empty queue*'.

Formally, given the system state $\mathbf{S}(t)$ during transmission frame t, a scheduling control $\mathbf{U}(t)$ is *feasible* if and only if the resulted withdrawal/insertion vectors satisfy the following feasibility constraints:

$$0 \leq W_{k,i}(t) \leq \mathbb{1}_{\{X_{k,i}(t)>0\}} \cdot C_{k,i}(t), \quad \forall k, i \neq 0, \tag{5}$$

$$\sum_{i=0}^{L_i} W_{k,i}(t) = 1, \qquad \forall k. \tag{6}$$

According to Constraint (5), a packet is withdrawn from a queue i in stack k only if this queue is connected and non-empty, i.e., $C_{k,i}(t) = 1$ and $X_{k,i}(t) > 0$. Constraint (6) insures that only one node in every set of nodes is allowed to transmit at any given transmission frame t. Define the set $\mathcal{U}(\mathbf{S}(t))$ as the set of all feasible scheduling controls $\mathbf{U}(t)$ when the system is in state $\mathbf{S}(t)$.

C. Policies for Dynamic Packet Scheduling

A *packet scheduling* policy π (or policy π for short) is a rule that determines the feasible control vectors $\mathbf{U}(t)$ for all t as a function of the past history and current state of the system, where the state history $\mathbf{H}(t)$ is given by the following sequence of random variables

$$\begin{aligned} \mathbf{H}(1) &= (\mathbf{X}(1)), \quad \text{and for} \quad t \geq 2: \\ \mathbf{H}(t) &= (\mathbf{X}(1), \mathbf{C}(1), \mathbf{A}(1), \ldots, \mathbf{C}(t-1), \mathbf{A}(t-1), \mathbf{C}(t)) \end{aligned} \tag{7}$$

Let $\mathcal{H}_t$ be the set of all state histories up to time frame t. Then a policy π can be formally defined as the sequence of measurable functions

$$g_t : \mathcal{H}_t \longmapsto \mathcal{Z}_+^{2K-1}, \text{ s.t. } g_t(\mathbf{H}(t)) \in \mathcal{U}(\mathbf{S}(t)), t = 1, 2, \ldots \tag{8}$$

where $\mathcal{Z}_+$ is the set of non-negative integers.

The set of feasible[5] scheduling policies described in Equation (8) is denoted by Π. We are interested in a subset of Π that we will introduce in the next section, namely the class of *Most Balancing* (MB) policies. We intend to prove the optimality of MB policies among all policies in Π.

[5]We say that a policy π is feasible if it selects a feasible scheduling control $\mathbf{U}^\pi(t) \in \mathcal{U}(\mathbf{S}(t))$ for all t.

III. The Class of MB Policies (Π^{MB})

In this section, we provide a description and mathematical characterization of the class of MB policies. Intuitively, the MB policies attempt to balance the sizes (leftover) of the queues for everyone of the K sets in the system. This can be achieved by minimizing the queue length differences for all sets of queues, at every time frame t.

We present next a more formal characterization of MB policies. We first define the '*imbalance index*' ($\kappa(\mathbf{x})$) of a vector $\mathbf{x}$.

Let $\mathbf{x} \in \mathcal{Z}_+^M$ be an M-dimensional vector. The imbalance index of $\mathbf{x}$ is defined as follows:

$$\kappa(\mathbf{x}) : \mathcal{Z}_+^M \longmapsto \mathcal{Z}_+, \quad \kappa(\mathbf{x}) = \sum_{i=1}^{M-1} \sum_{j=i+1}^{M} (x_{[i]} - x_{[j]}), \tag{9}$$

where $[k]$ denotes the index of the k^{th} longest component in the vector $\mathbf{x}$.

The above definition ensures that the differences are non-negative and a pair of components is accounted for in the summation only once. We define next the "*balancing interchange*" for the vector $\mathbf{x}$. We use this operation in the proof for the optimality of MB policies.

Definition: Balancing Interchange: Given vectors $\mathbf{x}, \mathbf{x}^* \in \mathcal{Z}_+^M$, we say that $\mathbf{x}^*$ is obtained from $\mathbf{x}$ by performing a *balancing interchange* if the two vectors differ in two components $i > 0$ and $j \geq 0$ only, where

$$x_i^* = x_i - 1, \quad x_j^* = x_j + 1, \quad s.t. \quad x_i \geq x_j + 1. \tag{10}$$

If the vector $\mathbf{x}$ represents a queue sizes vector then performing a balancing interchange would involve removing one packet from a larger queue i and adding it to a smaller queue j. We will show later (in Lemma 2) that such an operation will decrease the vector's imbalance index.

Given a state $\mathbf{s}(t)$ and a policy π that chooses the feasible scheduling control $\mathbf{u}(t) \in \mathcal{U}(\mathbf{s}(t))$ at transmission frame t; define the "updated" queue size, $\hat{x}_{k,j}(t)$, as the size of queue j in stack k after applying the control $\mathbf{u}(t)$ at slot k (and before adding the exogenous arrivals for stack 1, the SS nodes) during transmission frame t. Note that because we let $a_0(t) = w_{1,0}(t)$, $\hat{x}_{1,0}(t)$ may be negative. The updated queue sizes can be stated as follows:

$$\begin{aligned} \hat{x}_{1,i}(t) &= x_{1,i}(t) - w_{1,i}(t), \quad i \in \mathcal{L}^1, \quad and, \\ \hat{x}_{k,i}(t) &= x_{k,i}(t) + v_{k,i} - w_{k,i}(t), \quad k \geq 2, i \in \mathcal{L}^k \end{aligned} \tag{11}$$

Note that $\hat{x}_{k,i}(t), i > 0$ is a non-negative quantity due to the feasibility constraint (5).

At any given time frame t, the imbalance indices for the updated queue length vector $\hat{\mathbf{x}}_k(t), \forall k$ is given by $\kappa(\hat{\mathbf{x}}_k(t))$. From Equation (9), it follows that the minimum possible value of the imbalance index for a $M+1$-dimensional vector $\mathbf{x}$ is equal to $M \cdot x_{[M]}$ which is indicative of a fully balanced system.

We denote by Π^{MB} the set of all MB policies in the system. The elements of Π^{MB} are defined as follows:

Definition: Most Balancing Policies: A *Most Balancing* (MB) policy is a policy $\pi \in \Pi$ that, at every $t = 1, 2, \ldots,$

chooses feasible scheduling control vector $\mathbf{u}(t) \in \mathcal{U}(\mathbf{s}(t))$ that minimizes the imbalance indices $\kappa(\hat{\mathbf{x}}_k(t)), \forall k$, i.e.,

$$\Pi^{MB} = \left\{ \pi \in \Pi : \bigcap_{k=1}^{K} \underset{\mathbf{u}(t) \in \mathcal{U}(\mathbf{s}(t))}{\operatorname{argmin}} \kappa(\hat{\mathbf{x}}_k(t)), \forall t \right\} \quad (12)$$

In Equation (12), K sets of policies are defined through the K argmin functions. Policies in the k^{th} set minimize the imbalance index for the k^{th} stack updated queue length vector. The intersection of all sets results in a set of policies that minimize the imbalance index for all vectors. We say that a policy has the "*MB property*" during time frame n, if it chooses a control that satisfies Equation (12) at $t = n$. Then a MB policy can be defined as the policy that has the MB property at every time frame.

The set Π^{MB} in (12) is well-defined and non-empty, since the minimization is over a finite set of controls. Furthermore, the set of MB policies may have more than one element.

A. MB Policy Implementation

In this section, we provide a low-complexity algorithm to implement MB policies. This algorithm is defined next:

Definition: Algorithm 1: For every time frame t, Algorithm 1 selects the feasible control vector $\mathbf{u}(t)$ such that for every k, $u_{k,1}(t)$ is the longest connected queue in stack k, $u_{k,2}(t)$ is the shortest queue in stack $k+1$. That is

$$u_{k,1}(t) = l : l \in \underset{i \in \mathcal{L}_k : c_{k,i}(t)=1}{\operatorname{argmax}} (x_{k,i}(t) + v_{k,i}(t)) \quad (13)$$

$$u_{k,2}(t) = s : s \in \underset{i \in \mathcal{I}}{\operatorname{argmax}}\ c_{k+1,i}(t), \quad where$$

$$\mathcal{I} = \underset{j \in \{1,\ldots,L_{k+1}\}}{\operatorname{argmin}} x_{k+1,j}(t), \quad (14)$$

for all k, where $\mathbf{v}_k(t)$ is the insertion vector (endogenous packets) to stack k at time frame t. For queue 0 of each stack k we add one extra condition for the sake of mathematical accuracy, that is: "*If* $u_{k,1}(t) = 0$ *then* $u_{k,2}(t) = 0$." This may happen when the controller is forced to idle during any slot of frame t. □

Equation (14) identifies the shortest RS queue in stack $k+1$; if there are more than one queue that satisfy this condition, one of which (at least) is connected, then the connected one is the one selected as $u_{k,2}(t)$. Otherwise, $u_{k,2}(t)$ will be the shortest non-connected queue in the stack. The reason behind this extra condition is a special case where all the queues in stack $k+1$ have the same size, then queue $u_{k,2}(t)$ will be the longest queue in stack $k+1$ after adding the packet transmitted from queue $u_{k,1}(t)$. Selecting a connected queue in this case will provide the opportunity for the scheduler to select the longest queue as $u_{k+1,1}(t)$ at the next scheduling slot.

Lemma 1. *Algorithm 1 results in a feasible control vector* $\mathbf{u}(t)$ *for any* t.

Proof: According to Equation (13), packets are withdrawn from connected queues only. Furthermore, packets are withdrawn from the longest connected queue for any stack of queues. This will insure that as long as there is at least a single connected, non-empty queue in the stack then the longest connected queue will not be empty. Therefore, Equations (5) is satisfied. Furthermore, Equation (6) is satisfied by the definition of the scheduler control $\mathbf{u}(t)$. ■

Next we will prove a theorem that states that the policy resulted from the proposed implementation algorithm is indeed a MB policy. To do that we need the following lemma. Its proof can be found in [5].

Lemma 2. *Let* $\mathbf{x}$ *and* $\mathbf{x}^*$ *be two* $L+1$*-dimensional ordered vectors (in descending order); suppose that* $\mathbf{x}^*$ *is obtained from* $\mathbf{x}$ *by performing a balancing interchange of two components,* l *and* s*, of* $\mathbf{x}$*, where* $x_l > x_s$*, such that,* $s > l; x_l > x_a, \forall a > l$ *and* $x_s < x_b, \forall b < s$*. Then*

$$\kappa(\mathbf{x}^*) = \kappa(\mathbf{x}) - 2(s-l) \cdot \mathbb{1}_{\{x_l \geq x_s + 2\}} \quad (15)$$

Lemma 2 quantifies the effect of performing a balancing interchange on the imbalance index $\kappa(\mathbf{x})$ of the $L+1$-dimensional vector $\mathbf{x}$.

Theorem 1. *For the operation of the system under consideration, a MB policy can be constructed using Algorithm 1.*

We prove Theorem 1 by contradiction. We assume that a MB policy selects a control $\mathbf{u}(t)$ at t that does not satisfy Equations (13) – (14) for some k. The control vector selected by Algorithm 1 is feasible according to Lemma 1. Then using Lemma 2 we show that applying the controls selected by Equations (13) – (14) will result in an imbalance index $\kappa(\hat{\mathbf{x}}_k(t))$ that is smaller than that under the MB policy which contradicts Equation (12). Therefore, $\mathbf{u}(t)$ must satisfy Equations (13) – (14) for all k and the theorem follows.

Proof for Theorem 1: Given the system state $\mathbf{s}(t) = (\mathbf{x}(t), \mathbf{c}(t))$ at t; let l be the index of the longest connected queue in set k after adding the insertion vector $\mathbf{v}_k(t)$ (as in Equation (13)) and s be the index of the shortest queue in $k+1$ *before* executing the control $\mathbf{u}(t)$ that satisfies Equation (14). Let $\pi \in \Pi^{MB}$ be a MB policy that selects the scheduler control $\mathbf{u}(t) \in \mathcal{U}(\mathbf{s}(t))$ during frame t. To show a contradiction, we assume (to the contrary of Theorem 1) that $\mathbf{u}(t)$ does not satisfy Equations (13) – (14) for the scheduling slot $k \in \{1, \ldots, K\}$.

We show next that in this case, the control vector selected by Algorithm 1 during frame t will result in an imbalance index $\kappa(\hat{\mathbf{x}}_k(t))$ that is either (i) *less than* or (ii) *equal to* that obtained under a MB policy. Case (i) contradicts Equation (12); therefore, the MB policy must satisfy Equations (13) – (14). Case (ii) insures that Algorithm 1 satisfies Equation (12). In either case, Theorem 1 will follow.

Consider the following two cases corresponding to Equations (13) and (14):

1) $u_{k,1}(t)$ does not satisfy Equation (13) during slot k of frame t, i.e., $x_{k,u_{k,1}(t)}(t) + v_{k,u_{k,1}(t)}(t) < x_{k,l}(t) + v_{k,l}(t)$. Then $\hat{x}_{k,u_{k,1}(t)}(t) < \hat{x}_{k,l}(t) - 1$ (under π). According to Equation (10) we can perform a balancing interchange between components $u_{k,1}(t)$ and l that will reduce the imbalance index $\kappa(\hat{\mathbf{x}}_k(t))$. Therefore, π does not satisfy Equation (12) and hence it is not a MB policy. This contradicts the original assumption that $\pi \in \Pi^{MB}$. Therefore, we conclude that a MB policy must satisfy Equation (13). Note that

$x_{k,u_{k,1}(t)}(t) + v_{k,u_{k,1}(t)}(t) > x_{k,l}(t) + v_{k,l}(t)$ is not possible since queue $u_{k,1}(t)$ must be connected (feasibility constraint (5)) and queue l is the longest connected queue after adding the insertion vector by assumption. Equality is also not possible since $u_{k,1}(t)$ satisfies Equation (13) in this case.

2) $u_{k,2}(t)$ does not satisfy Equation (14) during slot k of frame t, i.e., $x_{k+1,u_{k,2}(t)}(t) > x_{k+1,s}(t)$. Then $x_{k+1,u_{k,2}(t)}(t) + v_{k+1,u_{k,2}(t)}(t) > x_{k+1,s}(t) + v_{k+1,s}(t) + 1$ (under π). Similar to the previous case, we can perform a balancing interchange between queues $u_{k,2}(t)$ and s. Again this will reduce the imbalance index $\kappa(\mathbf{x}_{k+1}(t) + \mathbf{v}_{k+1}(t))$. Therefore, π does not satisfy Equation (12) and hence it is not a MB policy. This contradiction leads us to conclude that π must satisfy Equation (14). Since s is the shortest queue by assumption, then $x_{k+1,u_{k,2}(t)}(t) < x_{k+1,s}(t)$ is not possible. However, if $x_{k+1,u_{k,2}(t)}(t) = x_{k+1,s}(t)$ s.t. $u_{k,2}(t) \neq s$; in this case, if $c_{k+1,u_{k,2}(t)}(t) = c_{k+1,s}(t)$ then $u_{k,2}(t)$ satisfies Equation (14) during time frame t. Otherwise, i.e., $c_{k+1,s}(t) > c_{k+1,u_{k,2}(t)}(t) = 0$, then $x_{k+1,u_{k,2}(t)}(t) + v_{k+1,u_{k,2}(t)}(t) = x_{k+1,s}(t) + v_{k+1,s}(t) + 1$. In this case, if $u_{k+1,1}(t) = s$ then $\hat{x}_{k+1,u_{k,2}(t)}(t) = \hat{x}_{k+1,s}(t) + 2$. A balancing interchange between queues $u_{k,2}(t)$ and s will reduce the imbalance index $\kappa(\hat{\mathbf{x}}_{k+1}(t))$. Therefore, π does not satisfy Equation (12) and hence it is not a MB policy. By contradiction π must satisfy Equation (14).

If on the other hand $u_{k+1,1}(t) \neq s$ then a policy that chooses either $u_{k,2}(t)$ or s while keeping $u_{k,1}(t)$ and $u_{k+1,1}(t)$ the same will result in the same imbalance index $\kappa(\hat{\mathbf{x}}_{k+1}(t))$. Since $\pi \in \Pi^{MB}$ by assumption, then Algorithm 1, in this case, will result in a policy that belongs to Π^{MB} as well.

We conclude that a MB policy $\pi \in \Pi^{MB}$ satisfies Equations (13) and (14) for all k and Theorem 1 follows. ■

Remark: The presented algorithm shows that it is feasible (with minimum efforts) to implement the MB policies. The details of such implementation in a real wireless system is out of the scope of this work. □

IV. Optimality of MB Policies

In this section we provide a proof for the optimality of the Most Balancing (MB) policies for the operation of the system presented earlier and shown in Figure 1. We start by defining a partial order that facilitates the comparison of the cost functions under different policies. We also define a class of cost functions for the optimality problem under investigation.

A. Definition of the Partial Order ($\preccurlyeq$)

For the proof of our results, we require a methodology that enables the comparison of the queue lengths under different policies. The idea is to define an order that we refer to as the "preferred order" to compare queue length vectors in each stack under different policies. We define the relation $\sqsubseteq$ on $\mathcal{Z}_+^M$ for some $M > 0$ as follows; we say that two vectors $\tilde{\mathbf{x}}$ and $\mathbf{x}$ are related via $\tilde{\mathbf{x}} \sqsubseteq \mathbf{x}$ if:

- S1- $\tilde{x}_i \leq x_i$ for all i (i.e., point wise comparison),
- S2- $\tilde{\mathbf{x}}$ is a 2-component permutation of $\mathbf{x}$; the two vectors differ only in two components i and j, such that $\tilde{x}_i = x_j$ and $\tilde{x}_j = x_i$, or
- S3- $\tilde{\mathbf{x}}$ is obtained from $\mathbf{x}$ by performing a *"balancing interchange"* as in Equation (10).

Now we are ready to define the "preferred order".

Definition: Define the *preferred order* ($\preccurlyeq$) as the transitive closure of the relation $\sqsubseteq$ on the set $\mathcal{Z}_+^M, M > 0$. □

The transitive closure of $\sqsubseteq$ on the set $\mathcal{Z}_+^M$ is the smallest transitive relation on $\mathcal{Z}_+^M$ that contains the relation $\sqsubseteq$ [20]. In other words, $\tilde{\mathbf{x}} \preccurlyeq \mathbf{x}$ if the vector $\tilde{\mathbf{x}}$ is obtained from $\mathbf{x}$ by performing a sequence of operations that involve reductions, permutations of two components and/or balancing interchanges.

B. The Class $\mathcal{F}$ of Cost Functions

We prove the optimality of MB policies for a class of cost functions $\mathcal{F}$ that we define next.

Denote by $\mathcal{F}$ the class of real-valued functions on the set $\mathcal{Z}_+^M$ that are monotone and non-decreasing with respect to the partial order $\preccurlyeq$ defined earlier. For any two vectors $\tilde{\mathbf{x}}, \mathbf{x} \in \mathcal{Z}_+^M$, a function $f \in \mathcal{F}$ if and only if

$$\tilde{\mathbf{x}} \preccurlyeq \mathbf{x} \Rightarrow f(\tilde{\mathbf{x}}) \leq f(\mathbf{x}). \tag{16}$$

From (16) and the definition of preferred order, it can be easily shown that the function $f(\mathbf{x}) = x_1 + x_2 + \cdots + x_M$ belongs to $\mathcal{F}$.

C. MB Policies Are Optimal

In what follows, we present a theorem stating the optimality of MB policies among all feasible policies and its proof. We use $Y \leq_{st} Z$ to denote the usual stochastic ordering for two real-valued random variables Y and Z [9]. We state that a policy $\sigma \in \Pi$ '*dominates*' another policy $\pi \in \Pi$ if

$$f(\mathbf{X}_k^{\sigma}(t)) \leq_{st} f(\mathbf{X}_k^{\pi}(t)), \quad \forall\ t = 1, 2, \ldots, \forall\ k, \tag{17}$$

and for all cost functions $f \in \mathcal{F}$; where $\mathbf{X}_k^{\alpha}(t), t = 1, 2, \ldots,$ is the queue lengths process for stack k under policy α.

Note that from Equation (16) and the definition of stochastic ordering, $\mathbf{X}_k^{\sigma}(t) \preccurlyeq \mathbf{X}_k^{\pi}(t)$ for all k, t and all sample paths in a suitable sample space[6], is sufficient for policy domination.

Let $\mathbf{X}_k^{MB}$ and $\mathbf{X}_k^{\pi}$ be the queue size processes of stack k under $\pi^{MB} \in \Pi^{MB}$ and an arbitrary policy $\pi \in \Pi$ respectively. We define the following subsets of the set Π:

(a) $\Pi_\tau \subseteq \Pi$, the set of policies that has the MB property during frames $t \leq \tau$, and are arbitrary for $t > \tau$.

(b) $\Pi_\tau^k \subseteq \Pi$, the set of policies that has the MB property during frames $t \leq \tau - 1$ and during $t = \tau$ chooses the controls $u_{i,1}(\tau)$ and $u_{i,2}(\tau)$ for all $i \leq k$ according to a MB policy and chooses arbitrary (may not be MB) controls for $k < i \leq K$ and for $t > \tau$.

It is trivially obvious that $\Pi = \Pi_0$. We can also see that $\Pi_n^k, \forall k = 1, \ldots, K$ and $n = 0, 1, \ldots$ form a monotone sequence of subsets, such that $\Pi_n \equiv \Pi_n^K \subseteq \Pi_n^{K-1} \subseteq \ldots \subseteq \Pi_n^1 \subseteq \Pi_{n-1}$. Furthermore, the set Π_n for any n is not empty, since it contains the MB policies. The set Π^{MB} can be defined

[6]The intended sample space is the standard one used in stochastic coupling, see [10] for details.

as $\Pi^{MB} = \bigcap_{n=1}^{\infty} \Pi_n$. To prove Theorem 2 below, we need the following lemmas. Their proofs are given in the appendix.

Lemma 3. *Given $\pi \in \Pi_{\tau-1}$, a policy $\tilde{\pi} \in \Pi_\tau^1$ that dominates π, in the sense of Equation (17), can be constructed.*

Lemma 4. *For any policy $\pi \in \Pi_\tau^h, h = 1, \ldots, K$, there exist a policy $\tilde{\pi} \in \Pi_\tau^{h+1}$ that dominates π.*

The above lemmas provide a methodology to construct a MB policy from any arbitrary policy π using stepwise improvements on the original policy while maintaining policy domination.

Theorem 2. *For the operation of the system described in Section II, a Most Balancing policy $\pi^{MB} \in \Pi^{MB}$ dominates any arbitrary policy $\pi \in \Pi$, i.e.,*

$$f(\mathbf{X}_k^{MB}(t)) \quad \leq_{st} \quad f(\mathbf{X}_k^{\pi}(t)), \quad \forall k, \forall\, t = 1, 2, \ldots, \quad (18)$$

and for all cost functions $f \in \mathcal{F}$.

Proof: The proof of Theorem 2 is conducted as follows: Starting from any arbitrary policy $\pi \in \Pi$, we construct a sequence of policies $\pi_1, \pi_2, \ldots$, using Lemmas 3 and 4 repeatedly[7], that satisfy the following: (a) π_1 dominates the original arbitrary policy π, (b) $\pi_n \in \Pi_n$, i.e., π_n has the MB property during time frames $t = 1, 2, \ldots, n$, and, (c) π_m dominates π_n for $m > n$.

By definition, π is an arbitrary policy; therefore, $\pi \in \Pi_0$. We construct a policy $\pi_{1,1} \in \Pi_1^1$ that dominates π according to Lemma 3. Using Lemma 4 we construct a sequence of policies $\pi_{1,i} \in \Pi_1^i, i = 2, \ldots, K$ each of which dominates the previous one according to Lemma 4. The last such policy $\pi_1 = \pi_{1,K} \in \Pi_1^K \equiv \Pi_1$ that has the MB property during transmission frame $t = 1$ and dominates π.

By repeating the above for frames $t = 2, 3, \ldots$ we obtain a sequence of policies $\pi_n \in \Pi_n, n = 2, 3, \ldots$ that satisfy (a) – (c) above, i.e., each subsequently constructed policy has the MB property for one extra frame (compared to the previous one) and dominates all the previously constructed policies as well as the original policy π.

Lets denote the limiting policy (as $n \longrightarrow \infty$) of the sequence of policies constructed earlier by π^*. Then π^* has the MB property at all time and therefore, $\pi^* \in \Pi^{MB}$. From the previous construction, we conclude that π^* dominates π_n, for all $n < \infty$ as well as the original policy π. The proof is complete since the initial policy $\pi \in \Pi$ is assumed to be arbitrary. ■

V. Conclusion And Suggestions For Future Work

In this paper, we investigated an optimal dynamic packet scheduling policy for multi-hop network with random connectivity. We modelled the K-hop network by K tandem sets of queues each of which represents the collection of nodes in a single hop. The random connectivity can be used to capture the stochastic nature of wireless channels. This model is suitable for studying dynamic scheduling in wireless relay networks. We proved, using coupling arguments, that *most balancing* policies dominate all other feasible policies in that they minimize, in stochastic ordering sense, a class of cost functions of the system queue lengths. One such function is the total number of packets in the system. We also provided an algorithm to construct the optimal scheduling policy. We proved that the suggested algorithm produces a most balancing policy for the system under investigation. Examples where this model can be applied include process scheduling in multi-stage operations with intermittent connectivity that can be found in operational research, software engineering (performance analysis and optimization) as well as computer networks (wireless relay networks and multi-hop wireless networks in general).

This work contribute to a theoretical platform that can be used to launch further research in the area of optimal packet scheduling in wireless networks. For future work we suggest the following extensions to the model: (i) the two-state model may be extended to finite-state Markov channel, (ii) the use of multiple servers per hop in the model presented in Figure 1, i.e., multiple packets can be scheduled at each of the K slots, (iii) the use of matrix connectivity per hop rather than the vector connectivity used here, (iv) the relaxation of the symmetry assumption we imposed on arrivals and connectivities of queues in each hop. It is worth noting that the problem resulted from adding such extensions will not be a trivial one.

Appendix

We use stochastic coupling arguments in the proof of Lemmas 3 and 4. A brief introduction to the coupling method is given next.

A. Stochastic Coupling

In order to compare probability measures on a measurable space, it is often possible and sometime rewarding to construct random elements (variables or processes) on a common probability space with these measures as their distributions, such that this comparison can be conducted in terms of the random elements rather than the probability measures. Such construction is often referred to as *stochastic coupling*[8] [10]. To put it differently, coupling refers to the joint construction of two or more random elements (in a convenient way) in order to deduce properties of the individual random elements. A formal definition of coupling of two probability measures on the measurable space $(E, \mathcal{E})$ is given below [10].

A random element in $(E, \mathcal{E})$ is a quadruple $(\Omega, \mathscr{F}, \mathbf{P}, X)$, where $(\Omega, \mathscr{F}, \mathbf{P})$ is the sample space and X is the class of measurable mappings from Ω to E (X is an E-valued random variable, s.t. $X^{-1}(B) \in \mathscr{F}$ for all $B \in \mathcal{E}$).

Definition: *A coupling of the two random elements $(\Omega, \mathscr{F}, \mathbf{P}, \mathbf{X})$ and $(\Omega', \mathscr{F}', \mathbf{P}', \mathbf{X}')$ in $(E, \mathcal{E})$ is a random element $(\hat{\Omega}, \hat{\mathscr{F}}, \hat{\mathbf{P}}, (\hat{\mathbf{X}}, \hat{\mathbf{X}}'))$ in $(E^2, \mathcal{E}^2)$ such that*

$$\mathbf{X} \stackrel{\mathscr{D}}{=} \hat{\mathbf{X}} \quad \text{and} \quad \mathbf{X}' \stackrel{\mathscr{D}}{=} \hat{\mathbf{X}}', \quad (A\text{-}1)$$

[7] To construct π_i starting from π_{i-1} we apply Lemma 3 once and Lemma 4 $K - 1$ times.

[8] A familiar (but more restrictive) definition of stochastic coupling is the construction of two stochastic processes such that their paths coincide after some *coupling epoch* T.

where $\stackrel{\mathscr{D}}{=}$ *denotes 'equal in distribution'*. Note that the original random elements need not be defined on a common probability space, i.e., they may not have a joint distribution. The couplings (the constructed copies), on the other hand, have a joint distribution.

Originally, stochastic coupling was used by mathematicians to prove properties for stochastic processes, e.g., establishing asymptotic stationarity of birth and death processes. Later on, stochastic coupling methods were used to prove optimality results for dynamic control of queueing systems. cf. [21], [22], [11], [12].

B. Proofs for Lemmas 3 and 4

To prove Lemmas 3 and 4, we show how $\tilde{\pi}$, as described in these lemmas, can be constructed while maintaining policy domination as stated in these lemmas. In both proofs we apply the coupling method as follows: For the scheduling policy π, let ω be a given sample path of the system state process. A new sample path, $\tilde{\omega}$ and a new policy $\tilde{\pi}$ are constructed. This construction is detailed in the proof below. In reference to the coupling definition (Equation (A-1)), the "coupled" processes of interest in Equation (A-1) will be the queue sizes $\hat{\mathbf{X}} = \{\mathbf{X}_k(n)\}$ and $\hat{\mathbf{X}}' = \{\tilde{\mathbf{X}}_k(n)\}$.

A feasible scheduling policy chooses a feasible control $\mathbf{U}(t) \in \mathcal{U}(\mathbf{S}(t))$ at every frame t. Using Equations (3) and (4), the new queue states $\mathbf{x}_k(\cdot)$ under π and $\tilde{\mathbf{x}}_k(\cdot)$ under $\tilde{\pi}$ for all k, are computed.

One can show the dominance of policy $\tilde{\pi}$ over π by proving that the following K relations are satisfied for all t.

$$\tilde{\mathbf{x}}_k(t) \preccurlyeq \mathbf{x}_k(t), \quad \forall k = 1, \ldots, K \tag{A-2}$$

A queue length vector $\tilde{\mathbf{x}}_k$ is preferred over $\mathbf{x}_k$ (i.e., $\tilde{\mathbf{x}}_k \preccurlyeq \mathbf{x}_k$) *if* one of the statements S1, S2 or S3 (in Section IV-A) holds.

Proof for Lemma 3: The proof of Lemma 3 is carried out in two steps (detailed in Part 1 and Part 2) as follows: Starting from an arbitrary policy $\pi \in \Pi_{\tau-1}$ and a sample path $\omega = (\mathbf{x}(1), \mathbf{c}(1), \mathbf{a}(1), \ldots)$, we construct the sample path $\tilde{\omega}$ and the policy $\tilde{\pi}$ (as stated by Lemma 3) for times up to $t = \tau$ (in Part 1) and for $t > \tau$ (in Part 2).

Part 1: We construct $\tilde{\omega}$ to coincide with ω for all $t < \tau$, i.e., $\tilde{\mathbf{a}}(t) = \mathbf{a}(t)$ and $\tilde{\mathbf{c}}(t) = \mathbf{c}(t)$ for all $t < \tau$. Furthermore, we construct $\tilde{\pi}$ such that $\tilde{\mathbf{u}}(t) = \mathbf{u}(t)$ for all $t < \tau$. Then the resulting queue lengths under both policies are the same, i.e., $\tilde{\mathbf{x}}(\tau) = \mathbf{x}(\tau)$.

At $t = \tau$, let $\tilde{\mathbf{c}}(\tau) = \mathbf{c}(\tau)$ and $\tilde{\mathbf{a}}(\tau) = \mathbf{a}(\tau)$. We construct $\tilde{\pi}$ at $t = \tau$ by selecting $\tilde{u}_{k,1}(\tau)$ and $\tilde{u}_{k,2}(\tau), \forall k$ as follows:

1- *Construction of* $\tilde{u}_{1,1}(\tau)$. We have the following two cases to consider:

(i) The scheduling control $u_{1,1}(t)$ satisfies Equation (13) at $t = \tau$, i.e., $u_{1,1}(\tau) = l : l \in \operatorname{argmax}_{i \in \mathcal{L}_1 : c_{1,i}(\tau)=1} x_{1,i}(\tau)$, since $v_{1,i}(t) = 0, \forall i, t$ by assumption. Then we set $\tilde{u}_{1,1}(\tau) = u_{1,1}(\tau)$. The resulting queue lengths $\tilde{\mathbf{x}}_1(\tau+1) = \mathbf{x}_1(\tau+1)$ (since $\tilde{a}(\tau) = a(\tau)$ by construction), property (S1) holds true for set 1 queue length vector and (A-2) is satisfied for $k = 1$ at $t = \tau + 1$.

(ii) The scheduling control $u_{1,1}(t)$ does not satisfy Equation (13) at $t = \tau$. Then we set

$$u_{1,1}(\tau) = l : l \in \operatorname*{argmax}_{i \in \mathcal{L}_1 : c_{1,i}(\tau)=1} x_{1,i}(\tau).$$

Given the construction of $\tilde{c}(\tau)$ and $\tilde{a}(\tau)$, we conclude the following (we suppress the time argument for the subscript to simplify notation):

$$\hat{\tilde{x}}_{1,\tilde{u}_{1,1}}(\tau) = \hat{x}_{1,\tilde{u}_{1,1}}(\tau) - 1, \quad \hat{\tilde{x}}_{1,u_{1,1}}(\tau) = \hat{x}_{1,u_{1,1}}(\tau) + 1,$$
$$\textit{where} \quad \hat{x}_{1,\tilde{u}_{1,1}}(\tau) > \hat{x}_{1,u_{1,1}}(\tau) \tag{A-3}$$

Note that the selection of $\tilde{u}_{1,1}(\tau)$ and $u_{1,1}(\tau)$ have no affect on the remaining sets of queues, i.e., for $k > 1$.

From Equation (A-3) and the construction of the exogenous arrivals we conclude that property (S3) holds true for set 1 queue length vector and (A-2) is satisfied for $k = 1$ at $t = \tau + 1$.

2- *Construction of* $\tilde{u}_{1,2}(\tau)$ and $\tilde{u}_{2,1}(\tau)$. We have the following two cases to consider:

(i) The scheduling control $u_{1,2}(t)$ satisfies Equation (14) at $t = \tau$, i.e., $u_{1,2}(\tau) = s : s \in \operatorname{argmax}_{i \in \mathcal{I}} c_{1,i}(\tau)$, where $\mathcal{I} = \operatorname{argmin}_{j \in \{1,\ldots,K\}} x_{2,j}(\tau)$. Then we set $\tilde{u}_{1,2}(\tau) = u_{1,2}(\tau)$ and $\tilde{u}_{2,1}(\tau) = u_{2,1}(\tau)$. The resulting queue sizes for $k = 2$, $\tilde{\mathbf{x}}_2(\tau+1) = \mathbf{x}_2(\tau+1)$. Property (S1) holds true for the queue length vector of set 2 and (A-2) is satisfied for $k = 2$ at $t = \tau + 1$.

(ii) The scheduling control $u_{1,2}(t)$ does not satisfy Equation (14) at $t = \tau$. Then we set $u_{1,2}(\tau) = s : s \in \operatorname{argmax}_{i \in \mathcal{I}} c_{1,i}(\tau)$, where the set $\mathcal{I}$ is defined in case (i) above. We also set $\tilde{u}_{2,1}(\tau) = u_{2,1}(\tau)$. In this case and for all feasible selections of $u_{2,1}(\tau)$, the queue lengths of set 2 satisfy the following:

$$\tilde{x}_{2,\tilde{u}_{1,2}}(\tau+1) = x_{2,\tilde{u}_{1,2}}(\tau+1) + 1,$$
$$\tilde{x}_{2,u_{1,2}}(\tau+1) = x_{2,u_{1,2}}(\tau+1) - 1,$$
$$\textit{where } x_{2,\tilde{u}_{1,2}}(\tau+1) < x_{2,u_{1,2}}(\tau+1). \tag{A-4}$$

According to Equation (A-4), $\tilde{\mathbf{x}}_2(\tau+1)$ is obtained from $\mathbf{x}_2(\tau+1)$ by performing a balancing interchange of two components $\tilde{u}_{1,2}(\tau)$ and $u_{1,2}(\tau)$. Therefore, property (S3) holds true for queue length vector of set 2, and hence, Equation (A-2) is satisfied for $k = 2$ at $t = \tau + 1$.

3- *Construction of* $\tilde{u}_{k,i}(\tau)$ for $k > 2$. We set $\tilde{u}_{k,i}(\tau) = u_{k,i}(\tau), \forall k, i$. The resulting queue lengths satisfy $\tilde{\mathbf{x}}_k(\tau+1) = \mathbf{x}_k(\tau+1)$, property (S1) holds true for the queue length vector for all the remaining sets of queues and Equation (A-2) is satisfied for all $k > 2$ at $t = \tau + 1$.

The above concludes the construction of $\tilde{\pi}$ for $t = \tau$. As we can see from the construction steps 1 and 2 above, $\tilde{\pi}$ chooses the controls $\tilde{u}_{1,1}(\tau)$ and $\tilde{u}_{1,2}(\tau)$ similar to a MB policy. We also showed (in steps 1 – 3 above) that Equation (A-2) is satisfied for $t = \tau + 1$. This summarizes the construction of $\tilde{\pi}$ for all time frames $t \leq \tau$.

In Part 2 below, we construct $\tilde{\pi}$ for $t > \tau$. We also show using forward induction that Equation (A-2) is satisfied for all time frames $t > \tau$.

Part 2: In Part 1, we constructed $\tilde{\omega}$ and $\tilde{\pi}$ for $t \leq \tau$. We now do the same for $t > \tau$. The induction argument proceeds

as follows: We already showed (in Part 1 above) that either (S1), (S2) or (S3) is satisfied, and therefore Equation (A-2) is satisfied, for all k at $t = \tau + 1$. Let $\tilde{\pi}$ and $\tilde{\omega}$ be defined up to time $n - 1 \geq \tau$ and that either (S1), (S2) or (S3) is satisfied for all $\mathbf{x}_k(t)$ at $t = n$, which leads to $\tilde{\mathbf{x}}_k(n) \preccurlyeq \mathbf{x}_k(n), \forall k$. We will show that at time frame n, we can construct $\tilde{\pi}$ such that Equation (A-2) is satisfied at $t = n + 1$. To do that, we have to show that either (S1), (S2) or (S3) holds for $\mathbf{x}_k(t)$ at time frame $t = n + 1$.

The construction (for $t = n$) is carried out in K steps starting with $k = 1$. We utilize the symmetry of these K steps to reduce the construction efforts by iterating the same construction steps for $k = 2, 3, \ldots, K$.

Starting from $k = 1$, we consider three cases that correspond to properties (S1), (S2) and (S3) of the vector $\mathbf{x}_k(n)$. For each one of these cases, we consider three sub-cases that correspond to properties (S1), (S2) and (S3) of the vector $\mathbf{x}_{k+1}(n)$. For each of the last three sub-cases, we consider three sub-cases for the vector $\mathbf{x}_{k+2}(n)$, and so on.

1- $\tilde{\mathbf{x}}_1(n) \leq \mathbf{x}_1(n)$ (i.e., property (S1) holds for $\mathbf{x}_1(t)$). We set $\tilde{\mathbf{c}}_1(n) = \mathbf{c}_1(n)$ and $\tilde{\mathbf{a}}(n) = \mathbf{a}(n)$. We set $\tilde{u}_{1,1}(n) = u_{1,1}(n)$. The resulted queue lengths for set 1 satisfy (S1), i.e., $\tilde{\mathbf{x}}_1(n+1) \leq \mathbf{x}_1(n+1)$, and (A-2) holds for $k = 1$ at $t = n+1$. The controls $\tilde{u}_{1,2}(n)$ and $\tilde{u}_{2,1}(n)$ are constructed below and the queue lengths for $k = 2$ are computed as follows:

(a) $\tilde{\mathbf{x}}_2(n) \leq \mathbf{x}_2(n)$ (i.e., property (S1) holds for $\mathbf{x}_2(n)$). We set $\tilde{\mathbf{c}}_2(n) = \mathbf{c}_2(n)$. We also set the controls $\tilde{u}_{1,2}(n) = u_{1,2}(n)$ and $\tilde{u}_{2,1}(n) = u_{2,1}(n)$. In this case, (S1) is satisfied for $k = 2$ and (A-2) holds for $k = 2$ at $t = n + 1$.

(b) $\tilde{\mathbf{x}}_2(n)$ is a 2-component permutation of $\mathbf{x}_2(n)$ (i.e., property (S2) holds for $\mathbf{x}_2(t)$). Let queues i and j in set $k = 2$ be the indices of the two permuted queues. Then let $\tilde{c}_{2,i}(n) = c_{2,j}(n)$, $\tilde{c}_{2,j}(n) = c_{2,i}(n)$ and $\tilde{c}_{2,m}(n) = c_{2,m}(n), \forall m \neq i, j$. The controls $u_{1,2}(n)$ and $u_{2,1}(n)$ are constructed as follows:

$$\tilde{u}_{1,2}(n) = \begin{cases} i & \text{if } u_{1,2}(n) = j \\ j & \text{if } u_{1,2}(n) = i \\ m & \text{if } u_{1,2}(n) = m, \forall m \neq i, j \end{cases} \tag{A-5}$$

$$\tilde{u}_{2,1}(n) = \begin{cases} i & \text{if } u_{2,1}(n) = j \\ j & \text{if } u_{2,1}(n) = i \\ m & \text{if } u_{2,1}(n) = m, \forall m \neq i, j \end{cases} \tag{A-6}$$

From the construction of $\tilde{\pi}$, it is clear that property (S2) is satisfied again for $k = 2$ at $t = n + 1$ and so is (A-2) for $k = 2$.

(c) $\tilde{\mathbf{x}}_2(n)$ is obtained from $\mathbf{x}_2(n)$ by performing a balancing interchange as described by Equation (10) (i.e., property (S3) holds for $\mathbf{x}_2(t)$). Let i and j be the indices of the two queues in set $k = 2$ involved in the balancing interchange, such that $x_{2,i}(n) \geq x_{2,j}(n) + 1$. There are two cases to consider:

(i) $x_{2,i}(n) = x_{2,j}(n) + 1$. Then $\tilde{x}_{2,i}(n) = x_{2,j}(n)$ and $\tilde{x}_{2,j}(n) = x_{2,i}(n)$. This case corresponds to case (1b) above and the same construction of $\tilde{\omega}$ and $\tilde{\pi}$ apply. The resulted queue lengths for $k = 2$ at $t = n + 1$ will satisfy property (S2) and (A-2) holds for $k = 2$ as well.

(ii) $x_{2,i}(n) > x_{2,j}(n) + 1$. We set $\tilde{\mathbf{c}}_2(n) = \mathbf{c}_2(n)$ and $\tilde{u}_{1,2}(n) = u_{1,2}(n)$.

If "$c_{2,j}(n) = 1$, $c_{2,m}(n) = 0, \forall m \neq j$ and $x_{2,j}(n) \leq 0$"[9] (i.e., queue j is the only connected queue in set $k = 2$ which happens to be empty), then $u_{2,1}(n) = 0$ (i.e., forced idling) according to the feasibility constraint (5). Then we set $\tilde{u}_{2,1}(n) = j$, which is a feasible control since $\tilde{x}_{2,j}(n) = x_{2,j}(n) + 1$ according to Equation (10). The resulted queue length vector for $k = 2$ in this case satisfies (S1), i.e., $\tilde{\mathbf{x}}_2(n + 1) \leq \mathbf{x}_2(n + 1)$, and (A-2) holds for $k = 2$.

Else, i.e., $x_{2,j}(n) > 0$ and/or there are other connected queues in the stack, then we set $\tilde{u}_{2,1}(n) = u_{2,1}(n)$. This action preserve property (S3) for $k = 2$ and Equation (A-2) holds for $k = 2$ at $t = n + 1$.

The construction process for the remaining sets $k = 3, \ldots, K$ can be done iteratively by repeating steps (a) - (c) for each k and for each of the sub-cases above in a nested fashion. The same conclusion that Equation (A-2) holds for all $k > 2$ at $t = n + 1$ is reached.

This concludes the construction of $\tilde{\omega}$ and $\tilde{\pi}$ for case (1) during time frame $t = n$.

2- $\tilde{\mathbf{x}}_1(n)$ is a 2-component permutation of $\mathbf{x}_1(n)$ (i.e., property (S2) holds for $\mathbf{x}_1(t)$).

Let i and j be the indices of the two permuted SS queues (i.e., in set $k = 1$). Then let $\tilde{c}_{1,i}(n) = c_{1,j}(n)$, $\tilde{c}_{1,j}(n) = c_{1,i}(n)$ and $\tilde{c}_{1,m}(n) = c_{1,m}(n), \forall m \neq i, j$. Similarly, $\tilde{a}_i(n) = a_j(n)$, $\tilde{a}_j(n) = a_i(n)$ and $\tilde{a}_m(n) = a_m(n), \forall m \neq i, j$. We construct the control $u_{1,1}(n)$ as follows:

$$\tilde{u}_{1,1}(n) = \begin{cases} i & \text{if } u_{1,1}(n) = j \\ j & \text{if } u_{1,1}(n) = i \\ m & \text{if } u_{1,1}(n) = m, \forall m \neq i, j \end{cases} \tag{A-7}$$

From Equation (A-7), it is clear that property (S2) holds again for $\tilde{\mathbf{x}}_1(n + 1)$ and $\mathbf{x}_1(n + 1)$, and (A-2) is satisfied for $k = 1$ at $t = n + 1$. Analogous to case (1-) above, we consider three cases for the construction of $u_{1,2}(n)$ and $u_{2,1}(n)$ that correspond to (S1), (S2) and (S3) properties of the vector $\mathbf{x}_2(n)$. The construction of $\tilde{\omega}$ and $\tilde{\pi}$ in all three cases is analogous to that presented in cases (1a), (1b) and (1c) respectively, and the resulted queue length vector $\tilde{\mathbf{x}}_k(n + 1)$ satisfies (A-2) for all $k \geq 2$.

3- $\tilde{\mathbf{x}}_1(n)$ is obtained from $\mathbf{x}_1(n)$ by performing a balancing interchange as described by Equation (10) Let i and j be the indices of the two queues in set $k = 1$ involved in that operation, such that $x_{1,i}(n) \geq x_{1,j}(n) + 1$. We have the following cases to consider :

(i) $x_{1,i}(n) = x_{1,j}(n) + 1$. Then $\tilde{x}_{1,i}(n) = x_{1,j}(n)$ and $\tilde{x}_{1,j}(n) = x_{1,i}(n)$. This case corresponds to case (2-) above and the same construction of $\tilde{\omega}$ and $\tilde{\pi}$ apply and the same conclusions are reached. The resulted queue lengths at $t = n + 1$ will satisfy property (S2) and (A-2) follows.

(ii) $x_{1,i}(n) > x_{1,j}(n)+1$. We set $\tilde{\mathbf{a}}(n) = \mathbf{a}(n)$ and $\tilde{\mathbf{c}}_1(n) = \mathbf{c}_1(n)$.

If "$c_{1,j}(n) = 1$, $c_{1,k}(n) = 0, \forall k \neq j$ and $x_{1,j}(n) \leq 0$" (i.e., queue j is the only connected queue in set $k = 1$, which happens to be empty), then $u_{1,1}(n) = 0$ (i.e., forced idling). In this case, we set $\tilde{u}_{1,1}(n) = j$. This control is feasible since

[9] $x_{2,j}(n) < 0$ is possible since we did not exclude the dummy queue (j=0) from the argument.

$\tilde{x}_{1,j}(n) = x_{1,j}(n)+1$ according to Equation (10). The resulted queue lengths for $k = 1$ in this case satisfies (S1), i.e., $\tilde{\mathbf{x}}_1(n+1) \leq \mathbf{x}_1(n+1)$, and (A-2) is preserved at $t = n+1$.

Else, i.e., if $x_{1,j}(n) > 0$ and/or there are other connected queues in the stack, then we set $\tilde{u}_{1,1}(n) = u_{1,1}(n)$. This action preserve property (S3) and Equation (A-2) is satisfied at $t = n+1$.

In this case, as with the previous cases, there are three cases to consider for the construction of $u_{1,2}(n)$ and $u_{2,1}(n)$ which correspond to (S1), (S2) and (S3) properties of the vector $\mathbf{x}_2(n)$. The construction of $\tilde{\omega}$ and $\tilde{\pi}$ in each of these cases is analogous to the one presented in cases (1a), (1b) and (1c). The same conclusion that $\tilde{\mathbf{x}}_k(n+1)$ satisfies (A-2) for all $k \geq 2$ is reached.

The above concludes the construction of the policy $\tilde{\pi}$ at $t = n$, for $n > \tau$. We have shown that the constructed policy resulted in queue length vectors $\mathbf{x}_k(n+1), \forall k$ that satisfy Equation (A-2). Using induction argument we conclude that Equation (A-2) is satisfied for all t. By construction in Part 1, we have $\tilde{\pi} \in \Pi^1_\tau$. Furthermore, $\tilde{\pi}$ dominance over π follows from relation (16). ■

Proof for Lemma 4: We follow an argument similar to that we used in the proof of Lemma 3. The two proofs differ in Part 1, i.e., the policy construction for $t \leq \tau$. Part 2 of the proof is the same and will not be repeated.

Let π be an arbitrary policy such that $\pi \in \Pi^h_\tau$, where $h = 1, \ldots, K$; and let $\omega = (\mathbf{x}(1), \mathbf{c}(1), \mathbf{a}(1), \ldots)$. In the following, we will construct the sample path $\tilde{\omega}$ and the policy $\tilde{\pi}$ for times $t \leq \tau$.

For time $t < \tau$ we construct $\tilde{\omega}$ to coincide with ω, i.e., $\tilde{\mathbf{a}}(t) = \mathbf{a}(t)$ and $\tilde{\mathbf{c}}(t) = \mathbf{c}(t)$ for all $t < \tau$. We choose $\tilde{\pi}$ to be identical to π for all $t < \tau$, i.e., $\tilde{\mathbf{u}}(t) = \mathbf{u}(t)$. This results in similar queue lengths under both policies at $t = \tau$, i.e., $\tilde{\mathbf{x}}_k(\tau) = \mathbf{x}_k(\tau), \forall k$.

At time frame $t = \tau$, let $\tilde{\mathbf{c}}(\tau) = \mathbf{c}(\tau)$ and $\tilde{\mathbf{a}}(\tau) = \mathbf{a}(\tau)$. The construction of $\tilde{\pi}$ at $t = \tau$ is carried out as follows:

1- For all $k \leq h$, we set $\tilde{u}_{k,1}(\tau) = u_{k,1}(\tau)$ and $\tilde{u}_{k,2}(\tau) = u_{k,2}(\tau)$. Note that $u_{k,1}(\tau)$ and $u_{k,2}(\tau)$ in this case are most balancing controls by assumption ($\pi \in \Pi^h_\tau$). In this case, (S1) holds true for the queue length vector of sets $k \leq h$.

2- For all $k > h+1$, we set $\tilde{u}_{k,1}(\tau) = u_{k,1}(\tau)$ and $\tilde{u}_{k,2}(\tau) = u_{k,2}(\tau)$. Note that $u_{k,1}(\tau)$ and $u_{k,2}(\tau)$ may or may not be most balancing controls. Regardless of that, property (S1) holds true for the queue length vectors of sets $k > h+2$. We can not make any conclusions regarding the queue length vector for the set $h+2$ until we construct the control $\tilde{u}_{h+1,2}(\tau)$, which we present next.

3- For $k = h+1$, we construct $\tilde{u}_{k,1}(\tau)$ and $\tilde{u}_{k,2}(\tau)$ as follows:

(i) Construction of $\tilde{u}_{k,1}(\tau)$. If the scheduling control $u_{k,1}(\tau)$ satisfies Equation (13) at $t = \tau$, i.e., $u_{k,1}(\tau)$ is an MB control, then we set $\tilde{u}_{k,1}(\tau) = u_{k,1}(\tau)$. Property (S1) holds true for the queue length vector of set $k = h+1$ in this case.

If on the other hand, the scheduling control $u_{k,1}(\tau)$ does not satisfy Equation (13) at $t = \tau$. Then we set

$$\tilde{u}_{k,1}(\tau) = l : l \in \underset{i \in \mathcal{L}_k : c_{k,i}(\tau)=1}{\operatorname{argmax}} (x_{k,i}(\tau) + v_{k,i}(\tau)).$$

For set k, the queue lengths in this case satisfies the following (with some abuse of the notation):

$$\begin{aligned}
&\tilde{x}_{k,\tilde{u}_{k,1}}(\tau+1) = x_{k,\tilde{u}_{k,1}}(\tau+1) - 1,\\
&\tilde{x}_{k,u_{k,1}}(\tau+1) = x_{k,u_{k,1}}(\tau+1) + 1,\\
&\tilde{x}_{k,i}(\tau+1) = x_{k,i}(\tau+1), \quad \forall i \neq u_{k,1}(\tau), \tilde{u}_{k,1}(\tau),\\
&\textit{s.t.} \quad x_{k,\tilde{u}_{k,1}}(\tau+1) > x_{k,u_{k,1}}(\tau+1) \qquad \text{(A-8)}
\end{aligned}$$

From the definition of balancing interchange and Equation (A-8) above, we conclude that property (S3) is satisfied for set $k = h+1$.

(ii) Construction of $\tilde{u}_{k,2}(\tau), k = h+1$. If the scheduling control $u_{k,2}(\tau)$ satisfies Equation (14) at $t = \tau$, i.e., $u_{k,2}(\tau)$ is an MB control, then we set $\tilde{u}_{k,2}(\tau) = u_{k,2}(\tau)$. The resulted queue lengths for set $k+1 = h+2$ in this case satisfy (S1). Note that the control $\tilde{u}_{k,2}(\tau)$ does not affect the queue lengths of set $k = h+1$.

If on the other hand, the scheduling control $u_{k,2}(\tau)$ does not satisfy Equation (14) at $t = \tau$, then we set

$$\tilde{u}_{k,2}(\tau) = s : s \in \underset{i \in \mathcal{I}}{\operatorname{argmax}} \; c_{k+1,i}(\tau),$$

where $\mathcal{I} = \operatorname{argmin}_{j \in \{1,\ldots,L_{k+1}\}} x_{k+1,j}(\tau)$.

The resulted queue lengths for set $k+1$ satisfy the following:

$$\begin{aligned}
&\tilde{x}_{k+1,\tilde{u}_{k,2}}(\tau+1) = x_{k+1,\tilde{u}_{k,2}}(\tau+1) + 1,\\
&\tilde{x}_{k+1,u_{k,2}}(\tau+1) = x_{k+1,u_{k,2}}(\tau+1) - 1,\\
&\tilde{x}_{k+1,i}(\tau+1) = x_{k+1,i}(\tau+1), \quad \forall i \neq u_{k,2}(\tau), \tilde{u}_{k,2}(\tau),\\
&\textit{s.t.} \quad x_{k+1,\tilde{u}_{k,2}}(\tau+1) < x_{k+1,u_{k,2}}(\tau+1), \qquad \text{(A-9)}
\end{aligned}$$

the last inequality is a direct consequence of the selection of $\tilde{u}_{k,2}(\tau)$.

Using the definition of balancing interchange (Equation (10)) we conclude that property (S3) is satisfied for set $h+2$.

The above three cases covers all K sets of queues. The above concludes the construction of policy $\tilde{\pi}$ for time frame $t \leq \tau$. From the construction steps in case (3-) above, $\tilde{\pi}$ selects most balancing controls ($\tilde{u}_{k,1}(\tau)$ and $\tilde{u}_{k,2}(\tau)$) for all $k \leq h+1$ during time frame τ, i.e., $\pi \in \Pi^{h+1}_\tau$. We also showed that either (S1), (S2) or (S3) is satisfied for every case and therefore we conclude that Equation (A-2) is satisfied for all k at $t = \tau+1$.

The remaining part of the proof is to construct $\tilde{\pi}$ for $t > \tau$. Furthermore, starting from a preferred state at $t = \tau+1$, we must show that Equation (A-2) is satisfied for all $t > \tau$. This part is similar to Part 2 in the proof of Lemma 3, which is done using forward induction argument. We will not repeat it here to avoid redundancy.

The above provide a complete description of the policy $\tilde{\pi}$. This policy resulted in queue length vectors $\mathbf{x}(n+1), \forall k$ that satisfy Equation (A-2). Note that policy $\tilde{\pi} \in \Pi^{h+1}_\tau$ by construction in step (3-) above; its dominance over π follows from relation (16). q.e.d. ■

REFERENCES

[1] "MMR Harmonized Contribution on 802.16j (Mobile Multihop Relay) Usage Models," Document No. IEEE 802.16j-06/015, Sep. 2006.

[2] A. Sendonaris, E. Erkip and B. Aazhang, "User Cooperation Diversity–Part I: System Description," *IEEE Transactions on Communications*, vol. 51, no. 11, pp. 1927–1938, 2003.

[3] N. Laneman, D.N.C. Tse and G.W. Wornell, "Cooperative Diversity in Wireless Networks: Efficient Protocols and Outage Behaviour," *IEEE Trans. Inform. Theory*, vol. 50, pp. 3062–3080, 2004.

[4] A. Nosratinia, T.E. Hunter, A. Hedayat, "Cooperative Communication in Wireless Networks," *IEEE communications Magazine*, Vol. 42, pp. 74–80, Oct. 2004.

[5] H. Al-Zubaidy, C.C. Huang and J. Yan, "Dynamic Packet Scheduler Optimization in Wireless Relay Networks," to appear in *JSAC SI-CoNet*, Submitted/Accepted Mar. 2011/Aug. 2011.

[6] V. Stankovic, A. Host-Madsen, and Z. Xiong, "Cooperative diversity for wireless ad hoc networks," *IEEE Signal Processing Magazine*, vol. 23, pp. 37–49, Sep. 2006.

[7] S. M. Ross, *Stochastic Processes*. 2nd ed. New York: Wiley, 1996.

[8] Z. Roseburg, P. Varaiya, J. Walrand, "Optimal Control of Service in Tandem Queues," *IEEE Trans. Auto. Control.* AC27, pp. 600–610, 1982.

[9] D. Stoyan, *Comparison Methods for Queues and other Stochastic Models*. J. Wiley and Sons, Chichester, 1983.

[10] T. Lindvall, *Lectures on the coupling method.* New York: Wiley, 1992.

[11] L. Tassiulas and A. Ephremides, "Dynamic server allocation to parallel queues with randomly varying connectivity," *IEEE Trans. on Inf. Theory*, Vol. 39, Issue 2, pp.466–478, Mar. 1993.

[12] A. Ganti, E. Modiano and J. N. Tsitsiklis, "Optimal Transmission Scheduling in Symmetric Communication Models With Intermittent Connectivity," *IEEE Trans. on Inform. Theory,* Vol. 53, Issue 3, pp. 998–1008, Mar. 2007.

[13] H. Al-Zubaidy, I. Lambadaris and I. Viniotis, "Optimal Resource Scheduling in Wireless Multi-service Systems With Random Channel Connectivity," in *Proc. IEEE Globecom '09*, Honolulu, HI, USA, Dec. 2009.

[14] Y. Shi, W. Zhang, K. B. Latief, "Cooperative Multiplexing and Scheduling in Wireless Relay Networks," in *Proc. IEEE Int. Conf. on Commun.* (ICC 08), 2008.

[15] C.Y. Hong, A.C. Pang, "Link Scheduling with QoS Guarantee for Wireless Relay Networks," in Proc. *IEEE Conf. on Computer Commun.* (INFOCOM), Rio De Janeiro, Brazil, 2009.

[16] V. Sreng, H. Yanikomeroglu, and D. Falconer, "Relay selection strategies in cellular networks with peer-to-peer relaying," in *Proc. IEEE Vehic. Tech. Conf.*, pp. 1949–1953, Oct. 2003.

[17] M. Yu and J. Li, "Is amplify-and-forward practically better than decode-and- forward or vice versa?" in *Proc. IEEE Inter. Conf. Acoustics, Speech, and Signal Processing,* (ICASSP 05), vol. 3, pp. 365–368, Mar. 2005.

[18] A. Host-Madsen and J. Zhang, "Capacity bounds and power allocation for wireless relay channels," in *Proc. IEEE Trans. Information Theory,* vol. 51, no. 6, pp. 2020–2040, June 2005.

[19] M. Chen, S. Serbetli, and A. Yener, "Distributed power allocation for parallel relay networks," in *Proc. IEEE Global Telecom. Conf.* (GLOBECOM 05), vol. 3, pp. 1177–1181, Nov. 2005.

[20] R. Lidl and G. Pilz, *Applied abstract algebra*, 2nd ed., Undergrad. Texts in Math., Springer, 1998.

[21] J. Walrand, "A note on optimal control of a queuing system with two heterogeneous servers," *Systems and Control Letters*, Vol. 4, pp. 131–134, 1984.

[22] P. Nain, P. Tsoucas and J. Walrand, "Interchange arguments in stochastic scheduling", *Journal of Applied Probability*, Vol 27, pp. 815-826, 1989.