Energy-efficient Scheduling of Packet Transmissions over Wireless Networks

Abbas El Gamal, Chandra Nair, Balaji Prabhakar, Elif Uysal-Biyikoglu, and Sina Zahedi
Information Systems Laboratory
Stanford University
Stanford, CA 94305
abbas@isl.stanford.edu,mchandra@stanford.edu,balaji@stanford.edu, elif@stanford.edu,
szahedi@stanford.edu

Abstract— **The paper develops algorithms for minimizing the energy required to transmit packets in a wireless environment. It is motivated by the following observation: In many channel coding schemes it is possible to significantly lower the transmission energy by transmitting packets over a long period of time.**

Based on this observation, we show that for a variety of scenarios the offline energy-efficient transmission scheduling problem reduces to a convex optimization problem. Unlike for the special case of a single transmitter-receiver pair studied in [5], the problem does not, in general, admit a closed-form solution when there are multiple users. By exploiting the special structure of the problem, however, we are able to devise energy-efficient transmission schedules. For the downlink channel, with a single transmitter and multiple receivers, we devise an iterative algorithm, called MoveRight, that yields the optimal offline schedule. The MoveRight algorithm also optimally solves the downlink problem with additional constraints imposed by packet deadlines and finite transmit buffers. For the uplink (or multiaccess) problem MoveRight optimally determines the offline time-sharing schedule. A very efficient online algorithm, called MoveRightExpress, that uses a surprisingly small look-ahead buffer is proposed and is shown to perfom competitively with the optimal offline schedule in terms of energy efficiency and delay.

I. Introduction and Problem Formulation

The energy-efficiency of computing, signal processing and communication devices is key to the widespread deployment of wireless networks, especially of sensor and mobile ad hoc networks. On the networking side, several recent papers have proposed methods for conserving energy. For example, [1] proposes a randomized algorithm that allows nodes in a dense wireless network to switch between on and sleep modes so as to trade-off topology maintainence with energy conservation, [4] proposes a method for empirically measuring the energy consumed by a node in an ad hoc network by monitoring its power consumption, [5] considers the problem of minimizing the transmission energy of a wireless node and presents "lazy" schedules that trade-off delay for energy; and, [9] studies the problem of constructing energy-efficient multicast and broadcast trees.

This paper studies the problem of minimizing the energy required to transmit packets over a wireless network based on the following observation [5]: In many channel coding schemes, lowering transmission power and increasing the duration of transmission leads to a significant reduction in transmission energy. In particular, it was observed that for a given channel coding scheme if $w(\tau)$ is the energy expended for transmitting a packet over τ units of time, then $w(\tau)$ is a non-negative, monotonically decreasing, and strictly convex function of τ.

Before we introduce the minimum-energy scheduling problem, we briefly discuss it within the larger context of packet transmission protocols in wireless networks. Reducing energy consumption by lowering transmission power (and thus increasing transmission time) also reduces interference to other nodes, resulting in an increase in the overall throughput of the network. But, as noted in several previous papers ([10] is a recent reference), power control requires the participation of all nodes in the network: Nodes that reduce transmission power unilaterally risk suffering a high interference from nodes that do not. Thus, a network-wide protocol is needed to ensure that users adhere to the physical and link layer algorithms employed for energy minimization or for interference mitigation. While considerable research has been devoted to the design of good power control algorithms for dealing with interference, energy minimization is a more recent problem motivated by the advent of ad hoc and sensor networks. It is the goal of this paper to develop algorithms for energy-efficient scheduling in a wireless environment, building upon the approach taken in [5].

A. Minimum-Energy Transmission Scheduling Problem

For concreteness, consider the downlink channel in a wireless network involving a single transmitter and multiple receivers. Suppose that M packets arrive at the transmitter at random times t_i in the interval $[0, T]$ destined for one of n receivers. The node is required to transmit all M packets within the interval $[0, T]$[1]. Since the transmitter knows the destination of each packet, we may assume, without loss of generality, that the energy required to transmit packet i over τ units of time is given by the energy function $w_i(\tau)$. The $w_i(\tau)$ are assumed to satisfy the following conditions:

1. $w_i(\tau) \geq 0$.
2. $w_i(\tau)$ is monotonically decreasing in τ.
3. $w_i(\tau)$ is strictly convex in τ.
4. $w_i(\tau)$ is continuously differentiable and its derivative, $\dot{w}_i(\tau)$ tends to $-\infty$ as τ tends to 0.

The first three conditions have been justified in [5] by considering some channel coding schemes. The last condition is a technical condition introduced here for ease of exposition. It is not

[1]The imposition of a strict deadline, T, by which all transmissions had to terminate was intended to capture several realistic wireless scenarios (see [5] for further details).

required for the proofs, since strict convexity implies the existence of right and left derivatives and one can work with these. The last condition is also not artificial since it is satisfied by several channel coding schemes. For example, optimal coding over an additive white Gaussian noise (AWGN) channel with noise power N yields the energy function $\tau N(2^{\frac{2B}{\tau}} - 1)$ for a B-bit packet, which clearly satisfies condition 4.

Let s_i be the start time of the i^{th} packet's transmission and τ_i be its transmission duration. The *causality* constraint $s_i \geq t_i$ ensures that the transmission of a packet cannot begin before its arrival time. Even though it is not necessary for minimizing energy that packets be transmitted in the order of their arrivals, it is easy to see that any set of transmission times that satisfy the causality constraints and the overall deadline constraint T for some packet transmission order also satifies them when the packets are transmitted in the order of their arrivals. Thus, without loss of generality, we can assume that the s_i are monotonically increasing in i. With this assumption the deadline constraint requires that $s_M + \tau_M \leq T$. A vector of transmission times and transmission duration pairs, $\{(s_i, \tau_i), i = 1, ..., M\}$ that satisfies the above conditions will be called a *feasible* schedule. We are now ready to state the offline energy minimization problem.

Given:

a. a vector of packet arrival times $\{t_i, i = 1, ..., M\}$*, where* $t_1 = 0$*,* $t_i < t_{i+1}$*, and* $t_M < T$*, and*
b. energy functions $w_i(\tau)$ *which, for each* $i \in \{1, ..., M\}$*, satisfy the hypotheses 1-3 mentioned above;*

find a feasible schedule so as to minimize the total transmission energy: $\sum_{i=1}^{M} w_i(\tau_i)$.

We note that the convexity of the $w_i(\tau)$ makes this a convex optimization problem with linear constraints. For the special case of a single receiver, the $w_i(\cdot)$s are identical, say equal to the function $w(\cdot)$. In this case, the problem was solved explicitly in [5], yielding the following optimal offline schedule:

$$\tau_i^* = m_j \text{ if } k_{j-1} < i \leq k_j, \tag{1}$$

where m_j and k_j are obtained recursively as follows. Let $k_0 = 0$, and define

$$\begin{aligned} m_1 &= \max_{k \in \{1,...,M\}} \{\frac{t_{k+1}}{k}\} \text{ and} \\ k_1 &= \max\{k : \frac{t_{k+1}}{k} = m_1\}. \end{aligned}$$

For $1 \leq j \leq J$, let

$$\begin{aligned} m_{j+1} &= \max_{k \in \{1,...,M-k_j\}} \{\frac{t_{k_j+k+1} - t_{k_j+1}}{k}\} \text{ and} \\ k_{j+1} &= k_j + \max\left\{k : \frac{t_{k_j+k+1} - t_{k_j+1}}{k} = m_{j+1}\right\}, \end{aligned}$$

where $J = \min\{j : k_j = M\}$.

Unfortunately, for the general case involving multiple users, the convex energy minimization problem does not admit such an explicit solution. For example, in the downlink problem there is a significant difference: the $w_i(\cdot)$s are not all identical. This is because scheduling must be simulatneously done for the different channels between the transmitter and each receiver. These channels could possibly give rise to different packet transmission energy functions. For example, this occurs when the receivers are not equidistant from the transmitter. Since signal attenuation depends on the distance between the transmitter and each receiver, the energy required to transmit a packet reliably in time τ will be different for the different receivers.

This makes it impossible, in general, to obtain explicit solutions for the optimal offline minimum-energy schedule in terms of the $\{t_i\}$s as was possible before. Of course, one could use general convex optimization techniques to solve the above problem numerically. However, we note that the problem has special structure, making it amenable to special methods. In particular, its cost function is the sum of several convex energy functions, allowing us to perform local optimizations efficiently. Furthermore, the individual energy functions *decrease* monotonically, allowing local optimizations to be one-sided – namely, to the right. These special features are exploited in developing the MoveRight algorithm, which finds the optimal schedule efficiently.

The MoveRight algorithm also solves several other convex optimization problems related to determining offline energy efficient schedules in wireless networks. These include the following scenarios:

a. The downlink problem.
b. The optimal *time-sharing* schedule for the uplink multiaccess [2] problem.
c. All of the above scenarios when packets have individual deadlines before which they must be transmitted. The deadlines may be different for each packet, but must satisfy some conditions as stated later.
d. All of the above scenarios when the transmit buffer has a finite size of B.

Additionally, by employing a look-ahead buffer, the optimal offline schedule determined by the MoveRight algorithm can be used for online implementation. In this case, we show that a much faster version of the MoveRight algorithm, which we call MoveRightExpress, can be used to schedule the buffered packets. Of course, use of the look-ahead buffer would impose additional delays, but energy-efficiency requires one to trade-off an increase in delay for a decrease in energy consumption. The trade-off would be worth it if a small increase in delay leads to a significant reduction in energy. Previous work [5] shows that this is indeed the case for the single transmitter-receiver pair. In this paper we find that a small amount of look-ahead can lead to a substantial reduction in energy in the scenarios mentioned above.

[2] Recall that the uplink problem involves multiple users transmitting to one receiver using multiple access schemes. Information theory [2] tells us that time-sharing is not optimal for the general multiple access problem. We may nevertheless seek the optimal time-sharing schedule, similar to other work in the networking literature on the multiple access channel [6].

B. Organization of the paper

Section II develops the MoveRight algorithm for optimally solving the downlink offline transmission scheduling problem, and contains the main results of the paper. Section II-A provides the proof of optimality and Section II-B discusses the algorithm's worst-case complexity, implementation issues, and its fairness properties. Section II-C shows that MoveRight can also find the optimal offline schedule for scenarios involving deadlines for individual packets and finite transmit buffers. Section III discusses offline transmission scheduling for the uplink problem. Online scheduling using look-ahead buffers is presented in Section IV.

II. An Optimal Algorithm for the Offline Downlink Scheduling Problem

We develop the MoveRight algorithm for determining the optimal offline schedule for the downlink problem. After introducing the algorithm, establishing its optimality properties and analyzing its complexity, we shall show how it applies to other situations of interest.

Using notation introduced in the previous section, consider the problem of transmitting M packets that arrive at times $\{t_i, i = 1, ..., M\}$ during the period $[0, T]$, and as before, we assume $t_1 = 0$. For notational convenience, set $t_{M+1} = T$. Let s_i be the time the i^{th} packet starts transmitting and let τ_i be the duration of its transmission. A schedule is feasible if it is causal: $s_i \geq t_i$ for every i; and all packets are transmitted within the interval $[0, T]$: $\tau_1 + \cdots + \tau_M \leq T$. It is easy to see that $\tau_1 + \cdots + \tau_M = T$ is a necessary condition for the optimality of the transmission times $\{\tau_i\}$. Otherwise, we may simply increase some of the τ_i and reduce total energy (observe that increasing transmission times does not hurt the causality constraint). This reduces the causality constraint for all schedules which satisfy $\tau_1 + \cdots + \tau_M = T$ to $\sum_{i=1}^{j} \tau_i \geq t_{j+1}$.

We are required to find a feasible schedule so as to minimize the total transmission energy: $\sum_{i=1}^{M} w_i(\tau_i)$.

The MoveRight Algorithm: The main idea of the MoveRight algorithm is to iteratively move the starting times of packet transmissions to the right, one packet at a time, so that each move locally optimizes the overall energy function. As we shall see, this iterative local optimization leads to the globally optimum solution.

The algorithm proceeds iteratively. Initially, the start-times of all packets are set equal to their arrival times; that is, $s_i^0 = t_i, i = 1, \ldots, M$, and we set the transmission duration of packet i to $\tau_i^0 = s_{i+1}^0 - s_i^0$. Now consider the first two packets. Keeping $\tau_1^0 + \tau_2^0$ fixed, we move s_2^0 to s_2^1 (see Figure 1), where $s_2^1 \in [s_1^0, s_3^0]$ is the point which minimizes the sum of the transmission energies of the first two packets. Note that $s_2^1 \geq s_2^0$ necessarily, and therefore the start-time of packet 2 can only move to the *right*. In this simple case it is easy to see that leftward movements of the start time of packet 2 would violate the causality constraint, and are therefore not allowed. We prove that, in general, leftward movements are not necessary, and hence name the algorithm MoveRight.

Continuing, set τ_1^1 to be the transmission time of the first packet obtained after optimally increasing s_2^0 as above, and reset τ_2^0 by decreasing it by an amount $\tau_1^1 - \tau_1^0$.

Now consider the second and third packets. Again keeping $\tau_2^0 + \tau_3^0$ fixed, increase s_3^0 to s_3^1 optimally, and hence obtain τ_2^1. Reset τ_3^0 by reducing it by an amount $\tau_2^1 - \tau_2^0$, and proceed to obtain τ_i^1, for $i = 1, ..., M$. This completes the first *pass* of the algorithm. Continue to make additional passes and terminate the algorithm after pass K, where

$$K = \min\{k : \tau_i^k = \tau_i^{k-1}, \text{ for all } i, i = 1, ..., M\}.$$

A pseudo-code for the algorithm is given below.

```
k = 0;
flag = 0;
for  i = 1:M
   τ_i^0 = s_{i+1} − s_i;
end
while flag==0
   k=k+1;
   for i=1:M-1
      [τ_i^k, τ_{i+1}^k] =best([τ_i^{k-1}, τ_{i+1}^{k-1}, i, s_i^k]);
   end
   if τ^k == τ^{k-1}
      flag=1;
   end
end
```

Here $\texttt{best}([\tau_i^{k-1}, \tau_{i+1}^{k-1}, i, s_i^k])$ returns the optimal transmission durations when the total transmission duration is $\tau_i^{k-1} + \tau_{i+1}^{k-1}$ and the energy functions are $w_i(\cdot)$ and $w_{i+1}(\cdot)$. However, `best` also keeps in mind the causality constraint that $s_i^k + \tau_i^k \geq t_i^{k+1}$.

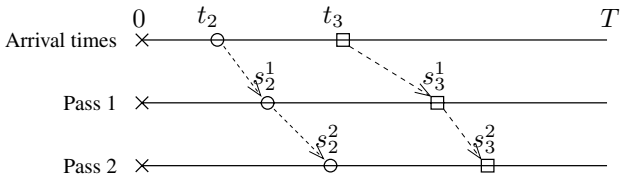

Fig. 1. Illustration of the MoveRight algorithm for 3 packets.

A. Proof of optimality

We first establish the following lemma in the absence of causality constraints.

Lemma 1: Consider two packets, 1 and 2, to be transmitted in the time interval $[s, t]$. Packet 1 is to begin its transmission at time s, while packet 2 is to end its transmission at time t. Let w_1 and w_2 be the transmission energy functions for packets 1 and 2, respectively, and assume that they satisfy conditions 1-4, then the following hold.

1. The optimal transmission times are unique.

2. Let $\hat{s}$ be the start time of the second packet's transmission in the optimal schedule. Then $\hat{s}$ increases when s increases,

holding t fixed. The same is also true if t increases and s is held fixed, and also if both s and t increase.

3. If the total time, $t-s$, decreases (increases) then the transmission durations of both packets decrease (increase, respectively).

Proof Let τ_1 and τ_2 be any transmission transmission schedule such that $\tau_1 + \tau_2 = t - s$.

1. Minimizing the strictly convex function $w_1(\tau_1) + w_2(t - s - \tau_1)$ over $0 \le \tau_1 \le t - s$ will yield the optimal schedule, which will obviously be unique given the strict convexity.

2. Consider the case when s increases to s' and t is fixed. Let τ_1^{opt} and τ_1^* denote the optimal transmission times for the first packet over intervals $[s,t]$ and $[s',t]$, respectively. Note that $\dot{w}_1(\tau_1^{opt}) - \dot{w}_2(t - s - \tau_1^{opt}) = 0$, where $\dot{w}$ denotes the first derivative.
Because the energy functions are strictly convex, their derivatives are strictly increasing. Therefore, since $s < s'$, it follows that $\dot{w}_1(\tau_1^{opt}) - \dot{w}_2(t - s' - \tau_1^{opt}) > \dot{w}_1(\tau_1^{opt}) - \dot{w}_2(t - s - \tau_1^{opt}) = 0$. Similarly, $\dot{w}_1(\tau_1^{opt} - (s' - s)) - \dot{w}_2(t - s - \tau_1^{opt}) < 0$.
We wish to find τ so that $\dot{w}_1(\tau) - \dot{w}_2(t - s' - \tau) = 0$. The above two statements and the uniqueness of the optimal value allow us to conclude that $\tau_1^{opt} - (s' - s) < \tau_1^* < \tau_1^{opt}$, or that $\tau_1^{opt} + s < \tau_1^* + s'$. This proves the claim.
The case when t increases can be established similarly. The last case can be handled by first increasing s and then increasing t.

3. Observe that the optimal transmission durations are just a function of total time available, $t - s$, and do not depend on the absolute values of s and t. Hence a decrease in $t - s$ can be made equivalent to increasing s to s', say, while keeping t fixed. From above we have $\tau_1^{opt} - (s' - s) < \tau_1^* < \tau_1^{opt}$. Since $\tau_1^* < \tau_1^{opt}$, we have that the transmission duration of the first packet decreases. For the second packet we need to show that $t - s - \tau_1^{opt} > t - s' - \tau_1^*$. This readily follows from $\tau_1^{opt} - (s' - s) < \tau_1^*$. The case when $t - s$ increases can be handled similarly. ■

We now introduce causality constraints to Lemma 1, which will be needed in the proof of Theorem 1. Note that with no causality constraints, the start-times are unconstrained and the energy-optimal start time of packet 2 can be to the *left* of (or earlier than) its arrival time. Of course, this can violate the causality constraint. However, it is not hard to see that, in this case, the optimal start-time for packet 2 is in fact equal to its arrival time. Thus, part 1 of Lemma 1 holds with causality constraints. Part 2 needs to be modified to:
*2. Let $\hat{s}$ be the start time of the second packet's transmission in the optimal schedule. Then $\hat{s}$ **does not decrease** when s increases, holding t fixed. The same is also true if t increases and s is held fixed, and also if both s and t increase.*

Now suppose there are M packets and let $\tau_1^k, \ldots, \tau_M^k$ be their transmission durations after the k^{th} pass of the MoveRight algorithm. Let $s_1^k = 0, s_i^k = \sum_{j=1}^{i-1} \tau_j^k$ for $i = 2, 3, ..., M$ and let $s_{M+1}^k = T$. Let $\tau_1^{opt}, \ldots, \tau_M^{opt}$ be the optimal transmission times, which exist because of the convexity of the problem and the compactness of the search space. Let $s_1^{opt} = 0, s_i^{opt} = \sum_{j=1}^{i-1} \tau_j^{opt}$ for $i = 2, 3, ..., M$ and let $s_{M+1}^{opt} = T = \sum_{j=1}^{M} \tau_j^{opt}$.

The main idea of the proof is to first show that, for each i, s_i^k is non-decreasing in k and that it is bounded above by s_i^{opt}. Therefore each $s_i^k \uparrow s_i^\infty$. We finish by establishing that $s_i^\infty = s_i^{opt}$, for every i.

Theorem 1: Let $s_i^k, s_i^\infty, s_i^{opt}$ be as defined before. Then
1. $s_i^k \le s_i^{k+1}$.
2. $s_i^k \le s_i^{opt}$.
3. $s_i^\infty = s_i^{opt}$.

Proof

1. Recall that the algorithm works in passes: For each fixed k, the algorithm determines s_i^k by increasing i from 1 through M. Because of the causality constraint, it follows trivially that $s_i^0 \le s_i^1$ for each $i, i = 1, 2, ..., M$ (recall that $s_i^0 = t_i$).
Suppose that $i' \ge 1$ and $k' \ge 1$ are the first time that there is a violation; that is, $s_{i'}^{k'} > s_{i'}^{k'+1}$. Since this is the first instance, we have that $s_{i'-1}^{k'} \le s_{i'-1}^{k'+1}$ and $s_{i'+1}^{k'-1} \le s_{i'+1}^{k'}$.
Consider the intervals $[s_{i'-1}^{k'}, s_{i'+1}^{k'-1}]$ and $[s_{i'-1}^{k'+1}, s_{i'+1}^{k'}]$. The first interval determined the boundaries within which the MoveRight algorithm would place $s_{i'}^{k'}$, the start-time of packet i' in the k'th pass. Likewise the second interval determines the boundaries for placing the start-time of packet i' in pass $k'+1$. The inequalities in the previous paragraph imply that each boundary point of the second interval is to the right of the corresponding boundary point in the fi rst interval. Given this, the modified version of part 2 of Lemma 1 implies $s_{i'}^{k'} \le s_{i'}^{k'+1}$. This contradicts the assumption $s_{i'}^{k'} > s_{i'}^{k'+1}$ and hence property (1) will always hold.

2. As above suppose that $i' \ge 1$ and $k' \ge 1$ are the first time that there is a violation; that is, $s_{i'}^{k'} > s_{i'}^{opt}$. (For reasons as above $k' = 0$ will not violate.)
Again, as before, we obtain $s_{i'-1}^{k'} \le s_{i'-1}^{opt}$ and $s_{i'+1}^{k'-1} \le s_{i'+1}^{opt}$. Notice that the boundary points of the interval $[s_{i'-1}^{k'}, s_{i'+1}^{k'-1}]$ are each to the left of the corresponding boundary points of $[s_{i'-1}^{opt}, s_{i'+1}^{opt}]$. Again by part 2 Lemma 1 we must have $s_{i'}^{k'} \le s_{i'}^{opt}$. This contradiction shows there can be no violation.

3. Let $\tau_i^\infty = \lim_{k\to\infty} \tau_i^k = s_{i+1}^\infty - s_i^\infty$. Note that the vectors $\{\tau_i^\infty\}$ and $\{\tau_i^{opt}\}$ are fixed points for the MoveRight algorithm: passing them once through the algorithm does not alter any entry. This is true of $\{\tau_i^\infty\}$, by definition. Since alterations by the MoveRight algorithm only result in energy reduction, the optimality of $\{\tau_i^{opt}\}$ ensures that it will be a fixed point. From part (2), we have $s_i^\infty \le s_i^{opt}$, for all $i = 1, \ldots, M+1$, with equality holding at both the boundaries. Also, from $s_{M+1}^\infty = s_{M+1}^{opt}$, we have $\sum_{i=1}^M \tau_i^\infty = T = \sum_{i=1}^M \tau_i^{opt}$.
We will argue by contradiction and hence let us assume that $j = \min\{i \ge 1 : \tau_i^\infty < \tau_i^{opt}, \tau_{i+1}^\infty \ge \tau_{i+1}^{opt}\}$. It is easy to see from the definition of j that, $s_{j+1}^\infty < s_{j+1}^{opt}$. Therefore, it follows from the feasibilty of s_{j+1}^∞ that the causality constraint did not play any role in the placement of s_{j+1}^{opt}. The same however cannot be said of s_{j+1}^∞. That is, the pairwise optimization of the transmission durations of packets j and $j+1$ could have yielded a start-time of s_{j+1}^* for packet $j+1$. However, packet $j+1$ was forced to begin transmission only at s_{j+1}^∞, due to causality constraints. It follows that $s_{j+1}^* \le s_{j+1}^\infty$.

Let $\tau_j^* = s_{j+1}^* - s_j^\infty$ and $\tau_{j+1}^* = s_{j+2}^\infty - s_{j+1}^*$. We have $\tau_j^* \leq \tau_j^\infty < \tau_j^{opt}$ and $\tau_{j+1}^* \geq \tau_{j+1}^\infty \geq \tau_{j+1}^{opt}$. Therefore,

$$\tau_j^* < \tau_j^{opt} \text{ and } \tau_{j+1}^* \geq \tau_{j+1}^{opt}. \quad (2)$$

We will now obtain the contradiction. First, suppose $\tau_{j+1}^* + \tau_j^* < \tau_{j+1}^{opt} + \tau_j^{opt}$. In this case from part 3 of Lemma 1 it follows that $\tau_j^* < \tau_j^{opt}$ and $\tau_{j+1}^* < \tau_{j+1}^{opt}$. Next suppose $\tau_{j+1}^* + \tau_j^* > \tau_{j+1}^{opt} + \tau_j^{opt}$. Then, by exactly similar arguments, it follows that $\tau_j^* > \tau_j^{opt}$ and $\tau_{j+1}^* > \tau_{j+1}^{opt}$. Finally, suppose $\tau_{j+1}^* + \tau_j^* = \tau_{j+1}^{opt} + \tau_j^{opt}$. Then, by part 1 of Lemma 1, it follows that $\tau_j^* = \tau_j^{opt}$ and $\tau_{j+1}^* = \tau_{j+1}^{opt}$. In all three cases we have contradicted equation (2) and proved the theorem. ∎

B. Properties of the MoveRight Algorithm

1. An ordering on arrival times: Because the algorithm moves start-times monotonically to the right, the worst-case inputs (packet arrival times) are easily identifiable in the following sense. Consider two different sets of arrival times, $\{t_i\}$ and $\{t_i'\}$, whose optimal schedules are identical. If $t_i \leq t_i'$, for every i, and s_i^k and ${s'}_i^k$ are the corresponding start-times after the k^{th} pass of the MoveRight algorithm, then $s_i^k \leq {s'}_i^k$, for every i and k. Therefore, when the MoveRight algorithm converges for the first input, it would have automatically converged for the second. We may therefore say that $\{t_i\}$ is worse than $\{t_i'\}$. This ordering can be used to determine the complexity of the algorithm, as described next.

2. Computational complexity: From part 2 of Theorem 1 we know that the MoveRight algorithm does not change the start-times of packets which are restricted by the causality constraint under the optimal schedule; that is, packets i such that $s_i^{opt} = t_i$. Call these the "immovable packets". The immovable packets have an interesting decoupling property: movements of packets to their left do not influence movements of packets to their right. Thus, the packets that move can be broken down into bands at whose end points there are immovable packets.

The rate of convergence of the MoveRight algorithm is determined by the rate at which packets in the slowest moving band will converge to their optimal positions. So, how fast does the slowest-moving band converge?

Observe that the start-times of packets within each band are not affected by the causality constraint. Therefore, their optimal start-times will be the same as determined by the MoveRight algorithm, assuming that the movable packets within a band all arrived at the beginning of the band! But, by the previous discussion on the ordering of arrival times, this last set of arrival times represents the worst-case as far as the convergence of the MoveRight algorithm is concerned.

Although the worst-case inputs are identified, without knowing the explicit form of the energy functions, it is difficult to bound the worst-case number of iterations of the MoveRight algorithm. However, assuming the energy functions are identical (the single receiver case), yields the following lemma, whose proof is presented in the Appendix.

Lemma 2: Suppose M packets with identical energy functions arrive at time 0 destined for a single receiver. Let s_i^k be the start-time of the i^{th} packet after the k^{th} pass of the MoveRight algorithm, and let $||s^k - s^{opt}|| = \max_i |s_i^k - s_i^{opt}|$. Then, given an $\varepsilon > 0$, $||s^k - s^{opt}|| < \varepsilon$ for $k \sim O\left(\frac{\log(\sqrt{M}/\varepsilon)}{\log(1/|\lambda_M|)}\right)$, where λ_M is the largest eigenvalue of the matrix exhibited in the Appendix.

Numerical evaluation of the above bound for values of M up to 1000 suggests growth rate of $M^{1.7}$ passes.

Simulation shows that the run time and number of iterations taken by the MoveRight algorithm are comparable (in terms of orders) when the energy functions are all identical, as compared with the case when they are distinct.

We considered 700 packets arriving at time 0, to be scheduled for transmission during $[0, 1000]$. Table I shows the number of moves, passes and the run-time of MoveRight when all 700 packets have equal energy functions. The algorithm was terminated when the total energy was within a certain percentage, denoted by % Opt in Table I, from its optimal value. Then we allowed each of the 700 packets to have an energy function chosen from a set of 10 types uniformly at random. The corresponding results are tabulated in Table II. The simulations were performed on a Pentium III 800 MHz machine.

% Opt.	No. of Passes	Run-time (sec)
10	85085	132.8
5	85609	133.4
1	86059	133.9
0.1	86164	134.0

TABLE I

THE NUMBER OF PASSES AND THE RUN-TIME OF MOVERIGHT FOR PACKETS WITH EQUAL ENERGY FUNCTIONS.

% Opt.	No. of Passes	Run-time (sec)
10	175425	240.1
5	186397	251.6
1	199081	264.9
0.1	203084	269.0

TABLE II

THE NUMBER OF PASSES AND THE RUN-TIME OF MOVERIGHT FOR PACKETS WITH DIFFERENT ENERGY FUNCTIONS.

3. Algorithm implementation: The main computational module in the execution of the MoveRight algorithm is the `best` routine, which involves just two individual energy functions. For each pair of energy functions, the `best` routine can be implemented via a precomputed lookup-table, resulting in significant speedup. Note that, by comparison, general convex optimization methods that do not exploit the special structure of the problem would need to perform a significant amount of computation at each iteration.

4. All packets available at the origin: An important special

case is when all of the M packets are available at $t = 0$. This situation is particularly relevant for the online implementation of the MoveRight algorithm via a look-ahead buffer, and for the discussion on fairness to follow next.

Observe that none of the M packets is constrained by causality: their start-times can be anywhere in $[0, T]$. Number the packets 1 through M and let $\tau_1, \ldots, \tau_M$ be the optimal schedule as determined by the MoveRight algorithm. We claim that any other numbering of the packets will also lead to each of them having the *same* transmission durations. To verify the claim, simply note that cost function we're minimizing is $\sum_i w_i(\tau_i)$ subject to the constraint $\sum_i \tau_i = T$. Given the strict convexity of the cost function (and hence the uniqueness of the optimal schedule), the solution of this problem is identical to the solution of the problem:

$$\text{Minimize: } \sum_i w_{\pi(i)}(\tau_{\pi(i)})$$
$$\text{subject to: } \sum_i \tau_{\pi(i)} = T,$$

for any permutation, π, of the numbers 1 through M.

5. Fairness: For concreteness, consider the downlink problem with two receivers. Suppose the transmit durations, $\{\tau_i\}$, of all packets are computed using the MoveRight algorithm. If, at the start of a new transmission, packets for both receivers are simultaneously present in the transmit buffer, then the packets may be transmitted in any order without affecting the energy-efficiency. This follows from the previous discussion point. Thus, when packets destined for different receivers are present in the buffer, in the interests of fairness, we may transmit packets in a round-robin fashion as opposed to a first-come-first-served order. The overall expenditure of energy is identical in both cases.

C. Extensions of the MoveRight Algorithm

Throughout this section we assume that there are M packets to be scheduled in an offline fashion, given the arrival times of the packets. We will show how the MoveRight algorithm can be used to arrive at the optimal offline schedule.

1. Packets have individual deadlines: Packet i, $i = 1, ..., M$, arrives at time t_i and must be transmitted by time $t_i + D_i$, where $D_i > 0$ is the deadline for packet i. Equally, if d_i is the departure time of packet i, then $d_i \leq t_i + D_i$. The D_i's are allowed to vary across packets. However, $t_i + D_i$, will be assumed to be monotonically increasing with i. Observe, that these impose additional *linear* constraints on the energy cost-function.

The only modification to make in the MoveRight algorithm is to change the `best` subroutine. The modified `best` subroutine simply takes into account the individual packet deadlines before returning the optimal transmission durations of two adjacent packets. It can be shown, but we omit it here due to lack of space, that the convergence and optimality properties are preserved under this modification.

2. Finite transmit buffers: Consider the downlink problem, where one transmitter is to send each of the M packets to one of n receivers. When the transmitter has a finite buffer of size, say B ($1 \leq B < M$), it is not allowed to simultaneously buffer more than B packets. (We include the packet currently being transmitted for determining the buffer occupancy at any time.)

A transmission schedule under the presence of a buffer of size B is valid if, and only if, for every i, packets i and $i + B$ never reside in the buffer simultaneously. This translates to the following constraint on the departure time: $d_i \leq t_{i+B}$. Rewriting the last constraint as $d_i \leq t_i + (t_{i+B} - t_i)$, we see that this is equivalent to the previous case when $D_i = t_{i+B} - t_i$. Note that if packets arrive in batches, then it is possible that the optimal schedule may be to set one or more transmission durations to 0 (thereby incurring infinite energy expenditure), if it is to satisfy the buffer constraint. This can be addressed either by dropping packets or by disallowing batch arrivals.

III. Offline Scheduling for the Uplink Problem

The uplink or multiaccess wireless channel consists of multiple transmitters and a single receiver. In general, users transmit simultaneously causing their signals to interfere at the receiver. The optimal rates at which the users can simultaneously transmit has been determined for fairly general channel models, *e.g.*, the Additive White Gaussian Noise (AWGN) channel (see Chapter 14 of [2]). The multiaccess offline scheduling problem involves the determination of time intervals and transmission rates obeying causality constraints. To make the discussion concrete we will assume the AWGN multiaccess channel model and restrict ourselves to two transmitters.

We assume that in time τ the first transmitter wishes to transmit a B_1-bit packet while the second transmitter wishes to transmit a B_2-bit packet. We let w_1 and w_2 be the received energies for users 1 and 2, respecively. Assuming receiver noise power N, it can be shown that w_1 and w_2 must obey the following conditions for some $\epsilon > 0$ for reliable communication to take place

$$\begin{aligned} w_1 &\geq \tau(N(2^{2B_1/\tau} - 1) + \epsilon) \\ w_2 &\geq \tau(N(2^{2B_2/\tau} - 1) + \epsilon) \\ w_1 + w_2 &\geq \tau(N(2^{2(B_1+B_2)/\tau} - 1) + \epsilon). \end{aligned}$$

Moreover, any (w_1, w_2) pair that satisfy these bounds can be achieved (with some probability of error that can be made as small as needed by proportionally increasing B_1, B_2, and τ) using simultaneous communication. Figure 2 plots the boundary of (w_1, w_2) pairs satisfying these conditions. If instead we restrict ourselves to *time-sharing* transmission schemes, where the users do not transmit simultaneously, we can only achieve (w_1, w_2) pairs satisfying

$$\begin{aligned} w_1 &\geq \alpha\tau(N(2^{2B_1/(\alpha\tau)} - 1) + \epsilon) \\ w_2 &\geq \bar{\alpha}\tau(N(2^{2B_2/(\bar{\alpha}\tau} - 1) + \epsilon), \end{aligned}$$

where $\alpha \in [0, 1]$ is the fraction of time τ the first user transmits and $\bar{\alpha} = 1 - \alpha$ is the fraction of time the second user transmits. The boundary of (w_1, w_2) pairs satisfying these conditions is also plotted in Figure 2. Note that the boundary of the time sharing region meets that of the optimal region at a single point.

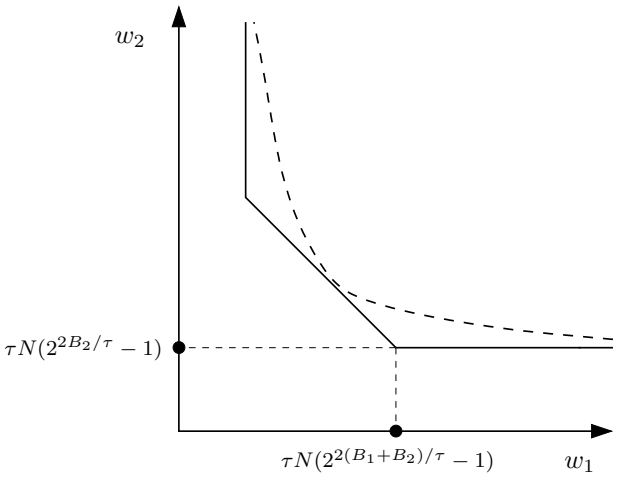

Fig. 2. Achievable (w_1, w_2) region for the AWGN multiaccess channel. The solid line represents the boundary of the optimal achievable region, while the dashed line represents the boundary of the region achievable using time-sharing.

The scheduling problem for the multiaccess channel involves the minimization of the total transmitted energy. First we discuss the problem of minimizing the energy needed to send two packets in time τ. Assuming path loss factors $a_1 > 0$ for user 1 and $a_2 > 0$ for user 2, the total transmitted energy can be expressed as $a_1 w_1 + a_2 w_2$. In the symmetric case, *i.e.*, when $a_1 = a_2$, it can be shown that time sharing achieves minimum total energy. Specifically the following lemma holds.

Lemma 3: For the AWGN multiaccess channel B_1 and B_2 can be reliably transmitted in time τ at total minimum energy using time sharing. In this case $w_1 + w_2 = \tau N(2^{2(B_1+B_2)/\tau} - 1)$.

This lemma can be used to show that a time-sharing multiaccess offline schedule exists that achieves minimum total energy. Such optimal time-sharing schedule can be obtained by simply merging the packets of the two users and using the optimal offline schedule for a single user. Unfortunately when $a_1 \neq a_2$, time sharing is no longer optimal for the offline multiacess scheduling problem. However, the optimal time-sharing offline schedule can be obtained by merging the packets and then applying the MoveRight algorithm.

We omit the proofs of Lemma 3 and the fact that time-sharing is optimal when $a_1 = a_2$ due to limited space.

IV. Online Scheduling

The offline version of the MoveRight algorithm lends itself nicely for online use by means of a look-ahead buffer. For concreteness, consider the downlink scheduling problem when there are K distinct receivers (and hence energy functions of K different types: $w_1, \ldots, w_K$). We are required to schedule packets arriving during the time interval $[0, T]$. Given the energy functions, the MoveRight algorithm provides the optimal offline schedule.

For an online implementation of the MoveRight algorithm, buffer all packets which arrive in the interval $[0, L]$, where $L \ll T$. Using the MoveRight algorithm, schedule these packets for departure in the interval $[L, 2L]$. Meanwhile, buffer the packets arriving in $[L, 2L]$ and schedule them for departure in the interval $[2L, 3L]$. Proceeding in this fashion, packets arriving in the interval $[(m-1)L, mL]$ are scheduled for departure using the MoveRight algorithm in the interval $[mL, (m+1)L]$. Call this scheme the "static look-ahead scheme". We shall now see that property 4 of the MoveRight algorithm vastly simplifies the scheduling complexity of the static look-ahead scheme, yielding the following algorithm.

The MoveRightExpress Algorithm: Suppose there are N packets in the look-ahead buffer at time mL, to be scheduled for transmission in the interval $[mL, (m+1)L]$. Let there be $n_1, \ldots, n_K$ packets destined for receivers 1 through K respectively. According to property 3 these packets may be scheduled in any order. Therefore, by reordering if necessary, we may assume the following order on the packets: all packets for receiver 1 appear first, followed by all packets for receiver 2, and so on, with the packets for receiver K appearing at the end.

Suppose that all packets are of equal length [3]. Since the energy functions of the first n_1 packets are all equal to w_1, we may assemble these packets into a "superpacket". The energy function of the superpacket is $W_1(\tau) = n_1 \, w_1(\frac{\tau}{n_1})$. Likewise assemble the packets for the other receivers into superpackets with corresponding energy functions $W_i(\tau) = n_i \, w_i(\frac{\tau}{n_i})$, $i = 2, \ldots, K$.

Now run the MoveRight algorithm on these K superpackets to obtain $\mathcal{T}_1, \ldots, \mathcal{T}_K$ as the optimal transmission durations. Given this and the fact that the packets destined for a single receiver must all have the same transmission duration (they have identical, strictly convex energy functions), it follows that the optimal transmission durations for the n_1 packets of receiver 1 are $\frac{\mathcal{T}_1}{n_1}$. Likewise, the optimal transmission durations for the n_i packets of receiver i are $\frac{\mathcal{T}_i}{n_i}$. Having determined the optimal schedule for all the packets, another application of property 4 implies that they may be transmitted in any order in the interval $[mL, (m+1)L]$.

Remark: It is worth noting the reduction in complexity achieved by the MoveRightExpress algorithm over the basic MoveRight algorithm. From depending on the total number of packets, M, the MoveRight Online algorithm's complexity only depends on the number of receivers, K.

As a comparison, we ran MoveRightExpress for the scenario of Table II. The results are tabulated in Table III. A comparison of Tables II and III shows that MoveRightExpress is much more efficient than the basic MoveRight algorithm. Again, the simulations were performed on a 800 MHz Pentium III machine.

In contrast, one could also consider the following "dynamic look-ahead scheme". Set the transmit time of the first packet, $\tau_1 = L$. Buffer all subsequent packets which arrive in the interval $[0, L]$. Schedule the second transmission using the MoveRight Online algorithm in the interval $[L, 2L]$. Suppose according to this schedule, the transmit time of the second packet is τ_2; i.e., it transmits from L to $L+\tau_2$. At $L+\tau_2$, we have access to all packets that arrived in the interval $[0, L+\tau_2]$. Given

[3] Extending this to variable-length packets is straightforward, see [8].

% Opt.	No. of Passes	Run-time (sec)
10	12	0.0
5	13	0.0
1	20	0.1
0.1	40	0.1

TABLE III

THE NUMBER OF PASSES AND THE RUN-TIME OF MOVERIGHTEXPRESS FOR PACKETS WITH DIFFERENT ENERGY FUNCTIONS, AS CONSIDERED IN THE SCENARIO OF TABLE II.

these packets, again using the MoveRight Online algorithm, schedule the third transmission in the interval $[L+\tau_2, 2L+\tau_2]$. Proceeding thus, we may schedule packets one at a time by dynamically taking new arrivals into account. If no new packets arrive and the buffer gets empty, then the next arrival is scheduled for a duration of L and the scheme proceeds as before.

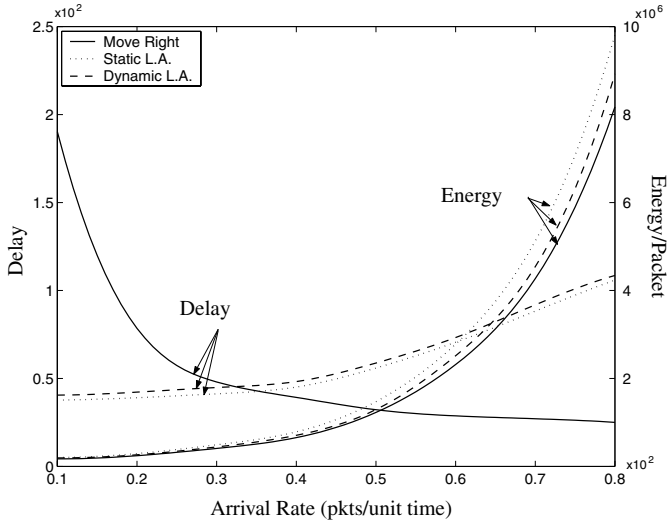

Fig. 3. Comparison of the Online Static and Dynamic Look-ahead schemes with the Offline MoveRight algorithm for a two-user downlink channel. The users' packets arrive according to two independent Poisson processes with identical rates. The energy functions used are $\frac{10^4}{6}\tau(2^{12/\tau}-1)$ and $\frac{16\times 10^4}{6}\tau(2^{12/\tau}-1)$, $T = 10000$, and the look-ahead window for each rate is chosen so that the energy/packet for static lookahead is 20% larger than that for the optimal offline. The simulated delay and energy/packet functions are plotted as a function of the combined arrival rate.

Of course, one expects the dynamic look-ahead scheme to outperform the static look-ahead scheme since it uses more information. However, the dynamic look-ahead scheme introduces considerable extra complexity, since it needs to run the MoveRight Online algorithm for *every* transmission. This is in contrast to the static look-ahead scheme, which only runs the MoveRight Online algorithm once for each look-ahead window of length L. This extra complexity would be worth it if the dynamic look-ahead scheme considerably outperforms the static look-ahead scheme. But, Figure 3 shows that the difference in energy and delay performance between the two schemes is negligible and quite competitive with respect to the offline algorithm when there are two receivers.

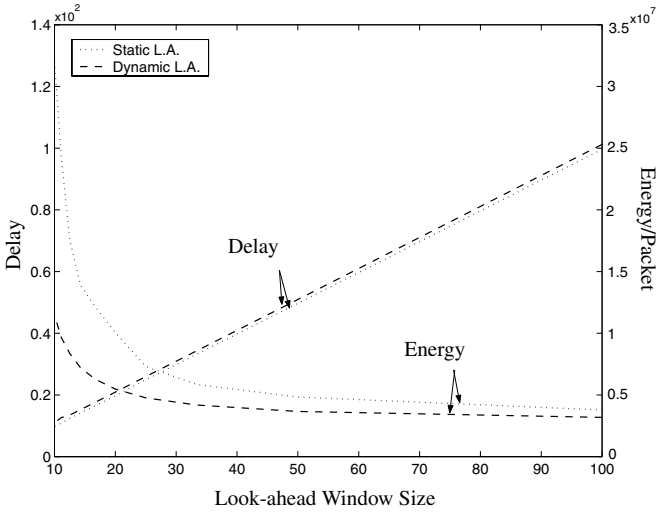

Fig. 4. Comparison of the Online Static and Dynamic Look-ahead schemes as the size of the look-ahead window increases. The packet generation, energy functions, and T used are the same as for Figure 3. The combined rate was 0.6packets/unit time. The MoveRight algorithm gives energy 2.5×10^6, and delay of 37.56.

Another interesting comparison is between the two online schemes mentioned above, as the size of the look-ahead window, L, varies. Clearly, larger values of L will make the online schemes compete better with the offline scheme in terms of energy, but will increase the delay considerably. On the other hand, small values of L will give good delay, but at the expense of energy efficiency. This suggests that there is a good choice for the size of the look-ahead window, L, which trades-off energy-efficiency and delay optimally for a given distribution of the arrival times. Figure 4 illustrates this trade-off when there are two users and the packet arrival times are independent Poisson processes. We notice that the energy curves have a sharp knee around $L = 20$, suggesting that most of the gain in energy-efficiency is obtained with a look-ahead window of this size.

Extension to Channels with fading: Suppose that the fading state of the channel (or channels) is known causally, at the end of each transmission to both the transmitter and receiver. Also suppose that the fading changes slowly compared to the packet transmission duration [4]. Knowing the fading state at time 0 is tantamount to knowing the energy functions of all packets (given this fade-state). With these assumptions, the dynamic look-ahead scheme described can be readily used: The transmission duration of the first packet is computed by running the MoveRightExpress algorithm with this set of energy functions. After the first packet is transmitted, the current fading state is used to compute the transmission duration of the second packet, and so on.

V. CONCLUSIONS

Recently, there has been a lot of research effort directed toward the design of low power signal processing and computing circuitry. On the networking side protocols are being designed

[4]These are standard assumptions for the slowly fading wireless channel in the literature (see, for example, [7]).

for minimum-energy routing, and for power control to mitigate interference.

We considered the energy-efficiency of packet transmission in several scenarios arising in wireless networks. For the downlink channel, we formulated the energy-efficient offline scheduling problem as a convex optimization problem and exploited its special structure to provide an efficient optimal algorithm, called MoveRight. We showed that MoveRight also optimally solves the downlink problem with additional constraints imposed by packet deadlines and finite transmit buffers. For the uplink (multiaccess) problem, MoveRight optimally determines the offline time-sharing schedule. A very efficient online algorithm, called MoveRightExpress, that uses a look-ahead buffer of small size was shown to perfom competitively with the optimal offline schedule in terms of energy efficiency and delay.

Further work consists of integrating the ideas developed in this paper with network-wide, decentralized, minimum-energy transmission protocols.

REFERENCES

[1] B. Chen, K. Jamieson, H. Balakrishnan, and R. Morris, "Span: An Energy-Efficient Coordination Algorithm for Topology Maintenance in Ad Hoc Wireless Networks," ACM Mobicom 2001, Rome, Italy, July 16-21.

[2] T. Cover, J. Thomas, *Elements of Information Theory*, Wiley Series in Telecommunications, John Wiley & Sons, 1991.

[3] A. El Gamal, E. Uysal-Biyikoglu and B. Prabhakar, "Reliable Communication with Minimum Energy," pre-print.

[4] L. Feeney and M. Nilsson, "Investigating the Energy Consumption of a Wireless Network Interface in an Ad Hoc Networking Environment," *Proc. IEEE Infocom 2001*.

[5] B. Prabhakar, E. Uysal-Biyikoglu and A. El Gamal. "Energy-efficient Transmission over a Wireless Link via Lazy Packet Scheduling". *Proc. IEEE Infocom 2001*.

[6] M.B. Pursley, H.B. Russell, J.S. Wysocarski, "Energy-efficient transmission and routing protocols for wireless multiple-hop networks and spread-spectrum radios," in *Information Systems for Enhanced Public Safety and Security.* IEEE/AFCEA EUROCOMM 2000. Page(s): 1 -5

[7] T. Rappaport, *Wireless Communication, Principles and Practice*, Prentice-Hall, Inc. N.J.,1996.

[8] E. Uysal-Biyikoglu, B. Prabhakar and A. El Gamal, "Energy-efficient Transmission over a Wireless Link via Lazy Packet Scheduling," Submitted to *IEEE Transactions on Networking*.

[9] J.E. Wieselthier, G.D. Nguyen, A. Ephremides, "On the construction of energy-efficient broadcast and multicast trees in wireless networks," *Proc. IEEE Infocom 2000*.

[10] M. Xiao and N. Shroff, E. K. P. Chong, "Utility-Based Power Control (UBPC) in Cellular Wireless Systems," *Proc. IEEE Infocom 2001*.

APPENDIX

CONVERGENCE PROPERTIES OF THE MOVERIGHT ALGORITHM

Here we provide a proof for Lemma 2 in Section II-B, which gives an estimate for the worst-case number of iterations for the convergence of the MoveRight algorithm. For this analysis it is assumed that the energy functions are identical and that all packets are available at time 0. The justifications for these assumptions were discussed in Section II-B. From [5], we know that the optimal scheduling times are equal.

Proof of Lemma 2: Let $\varepsilon > 0$ and s_i^k, for $i = 2, \ldots, M-1$ and $k \geq 1$, be the start-times of the packets at the beginning of the kth pass of the MoveRight algorithm, where $s_i^0 = 0$, for $i = 1, \ldots, M$, $s_1^k = 0$, and $s_{M+1}^k = T, \forall k \geq 0$. The algorithm is said to have ε-converged to the optimal solution if $\max_i(s_i^{opt} - s_i^k) < \varepsilon$.

Observe that the s_i^ks follow the recursion, $s_i^k = \frac{1}{2}(s_{i-1}^k + s_{i+1}^{k-1}), \forall k \geq 1, i = 2, \ldots M$. Let $s^k = [s_1^k s_2^k s_3^k s_4^k \cdots s_{M+1}^k]^t$, then the recursion can be rewritten as $s^k = \Gamma_M s^{k-1}$, where

$$\Gamma_M = \begin{bmatrix} 1 & 0 & 0 & 0 & \cdot & \cdot & \cdot & \cdot & \cdot & 0 \\ \frac{1}{2} & 0 & \frac{1}{2} & 0 & 0 & \cdot & \cdot & \cdot & \cdot & 0 \\ \frac{1}{4} & 0 & \frac{1}{4} & \frac{1}{2} & 0 & 0 & \cdot & \cdot & \cdot & 0 \\ \frac{1}{8} & 0 & \frac{1}{8} & \frac{1}{4} & \frac{1}{2} & 0 & 0 & \cdot & \cdot & 0 \\ \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ \frac{1}{2^{M-1}} & 0 & \frac{1}{2^{M-1}} & \frac{1}{2^{M-2}} & \cdot & \cdot & \cdot & \cdot & \frac{1}{4} & \frac{1}{2} \\ 0 & 0 & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & 0 & 1 \end{bmatrix}$$

Therefore, we have $\|s^\infty - s^k\|_\infty = \|s^\infty - \Gamma_M^k s^0\|_\infty = \|\Gamma_M^k(s^\infty - s^0)\|_\infty$, where $s^\infty = s^{opt} = [0 \ \frac{T}{M} \ \frac{2T}{M} \ \cdots \ T]^t$. This implies that, $s^\infty - s^o = [0 \ \frac{-T}{M} \ \frac{-2T}{M} \ \cdots \ \frac{-(M-1)T}{M} \ 0]^t$. Define Γ'_M as the matrix obtained by removing the first row, first column, last row and last column of Γ_M. Let $\tilde{s} = [\frac{-T}{M} \ \frac{-2T}{M} \ \cdots \ \frac{-(M-1)T}{M}]^t$. Now observe that $\|\Gamma_M^k(s^\infty - s^0)\|_\infty = \|{\Gamma'}_M^k \tilde{s}\|_\infty$.

Let λ_M be the largest eigen-value (in magnitude) of Γ'_M. Therefore, we have, $|\lambda_M|^k \|\tilde{s}\|_2 \geq \|{\Gamma'}_M^k \tilde{s}\|_2 \geq \|{\Gamma'}_M^k \tilde{s}\|_\infty$. Hence, for all k, such that $|\lambda_M|^k \|\tilde{s}\|_2 \leq \varepsilon$, we have $\|{\Gamma'}_M^k \tilde{s}\|_\infty \leq \varepsilon$. $\|\tilde{s}\|_2 = T\sqrt{\frac{M(M-1)(2M-1)}{6}} \sim T\sqrt{\frac{M}{3}}$. Taking T to be fixed, this implies for all $k \geq \frac{\log(\sqrt{\frac{M}{3}}\frac{T}{\varepsilon})}{\log(\frac{1}{|\lambda_M|})} \approx O\left(\frac{\log(\sqrt{M}\varepsilon^{-1})}{\log\left(\frac{1}{|\lambda_M|}\right)}\right)$, we have $\|{\Gamma'}_M^k \tilde{s}\|_\infty \leq \varepsilon$.

Below is a plot of $\frac{\log(\sqrt{\frac{M}{3}}\frac{T}{\varepsilon})}{\log(\frac{1}{|\lambda_M|})}$, for $T = 10000, \varepsilon = 0.1, M = 1, \ldots, 1000$.

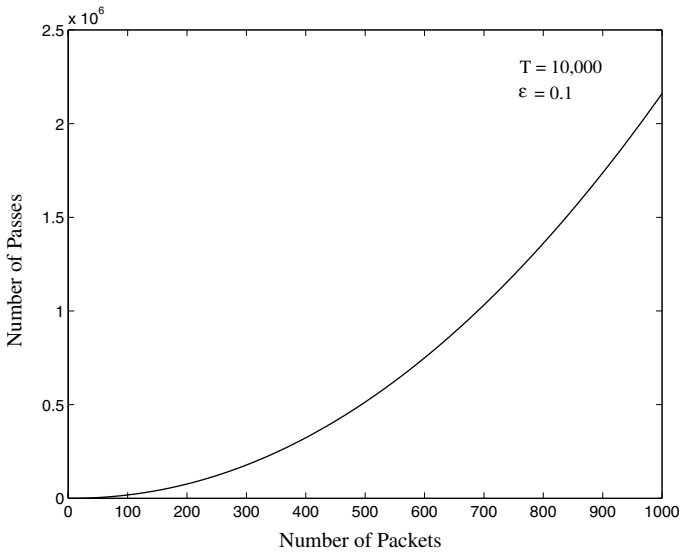

Fig. 5. Number of passes versus number of packets assuming $T = 10000$ and $\varepsilon = 0.1$.