Exploiting Mobility in Multi-hop Infostation Networks to Decrease Transmit Power

Furuzan Atay & Christopher Rose
WINLAB, Rutgers University
73 Brett Rd.,Piscataway, NJ 08854-8060
Email: furuzan/crose@winlab.rutgers.edu

***Abstract*— Mobility, rather than being a liability, can be an asset. If delay constraints are loose, it is possible for a given packet to observe many different network topologies as nodes move relative one another, and these different topologies can be treated as diversity. Opportunistic strategies can exploit these large scale changes in the channel quality to decrease transmit power at the expense of greater delay. We study the tradeoff between mobility, transmit power and delay and along the way develop simple greedy (packet-based) threshold rules for packet transmission.**

I. Introduction

In generic ad hoc networks, many applications have strict delay constraints requiring the network to be connected most of the time. However, a mobile system based on the infostations architecture [1], [2] targets applications with loose delay constraints and high data rate requirements. Thus, intermittent connectivity is both tolerated and expected.

Until recently, mobility was seen as undesirable since it complicates routing and can cause packet loss owing to intermittent node connectivity. However, if packets have loose delay constraints, it is possible for a given packet to observe many different network topologies as nodes move relative one another, and these different topologies can be treated as diversity [3], [4]. Thus, a packet can travel between nodes when the conditions are favorable and sojourn at a relay node in the meantime. That is, relay nodes need not forward a packet as soon as it is received. If the next hop is too costly (in terms of some suitable network resource), packets can be retained until the next good transmission opportunity.

Battery power is an important resource in all wireless networks. Although multi-hop transmission significantly reduces the power required to route a packet, decreasing transmit power is still important to prolong battery lifetime. Reducing transmit power also has and equally important "social" advantage. Less transmit power means less interference to the rest of the network. This usually results in a higher signal to interference and noise ratio (SINR) at receiver nodes and therefore higher data rates [5], [6].

In this paper, we study the trade-off between transmit power and delay in multi-hop infostation networks. We formulate and solve greedy local optimizations from the perspective of individual packets seeking carriage in the network – a *packet-eye view* protocol. In a companion paper, these strategies are applied to networks where packets can interfere with one another via interference and queueing delay [7].

II. Problem Statement

In problems where there are multiple objectives (like minimum power for given delay or minimum delay for given power consumption), a common approach is to define a cost structure reflecting all design concerns and to evaluate optimal operating points by varying the weight of cost structure components. We take a similar approach in that we consider a single packet in a network of mobile infostations whose destination is also mobile. We seek strategies that deliver the packet to its destination within some deadline T or earlier and minimize its cost.

We assume discrete intervals of duration δ during which packets can move directly between two nodes or stay put to wait for more opportune moments afforded by node mobility. Time is measured in integer units of δ. The cost, $c_{ij}(t) = c_{ji}(t)$, of transmissions between nodes i and j is a function of time owing to node mobility – N independent nodes in planar Brownian motion. To avoid boundary effects both x- and y-axes are wrapped around forming a torus. For all experiments we choose $\delta = 0.010\bar{a}^2/D$ where $\bar{a}$ is the average internodal distance and D is the diffusion coefficient. This choice allows significant, but not dramatic, channel variation between time steps.

We examine a lightly loaded system so that both multi access interference (MAI) and the queuing delay at nodes are assumed negligible. The components of cost are the power necessary for successful transmission above the ambient noise and the delay incurred during link traversals.

Achieving a target signal to interference/noise ratio (SINR), γ^*, at the receiver is assumed sufficient for successful transmission at some fixed rate R. The received power $P_j^{(r)}$ at node j due to a transmission from node i with power P_i is

$$P_j^{(r)} = P_i \left(\frac{d_{ij}}{d_0}\right)^{\alpha} \tag{1}$$

where d_{ij} is the distance from node i to j, $d_0 \leq d_{ij}$ is some minimum distance, and α is the propagation exponent. Accordingly, the minimum transmit power required for successful transmission is given by:

$$P_{ij}^* = N_0 W \gamma^* \left(\frac{d_{ij}}{d_0}\right)^{\alpha} \tag{2}$$

where W is the available bandwidth and N_0 is the background noise spectral intensity. We assume that the transmitter cannot transmit with arbitrarily small power and the minimum possible power level is equal to P_{ij}^* of distance $d_{ij} = d_0$.

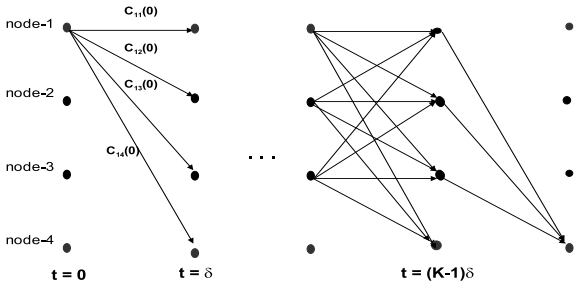

Fig. 1. A graph depicting all possible tours through a network of four nodes assuming packet origination at node 1 and forced termination by time $t = T$ at node 4. Link costs are shown for the first hop. Since 4 is the destination node, it is always a terminal node wherever it appears in any tour.

The power cost is simply defined as the minimum power used in a transmission,

$$C_P = N_0 W \gamma^* \left(\frac{d_{ji}}{d_0}\right)^{\alpha} \tag{3}$$

Likewise the delay cost is

$$C_D = \delta \tag{4}$$

and is assumed constant for each link traversal. Thus, the total cost for a transmission from node i to node j can be defined as

$$c_{ij}(t) = w_d \delta + w_p N_0 W \gamma^* \left(\frac{d_{ij}(t)}{d_0}\right)^{\alpha} \tag{5}$$

where w_d and w_p are positive weighting constants assigned to delay and power cost respectively.

Given link costs, packet motion through the network can then be modeled as a graph such as that depicted in FIGURE 1. Note that d_{ii} is assumed zero. The cost of a given tour is the sum of costs for the links traversed. Formally, if we denote a tour $\mathcal{T}$ by a sequence of integers $I_{\mathcal{T}} \equiv \{i_1, i_2, \cdots, i_K\}$ then

$$C_{\mathcal{T}} = \sum_{k=1}^{K} c_{i_{k-1} i_k}(k-1) \tag{6}$$

Finally, we note that the individual $c_{ij}(t)$ could be either deterministic or represent snapshots of a random process driven by the node mobility and other external time-varying processes.

Our goal is to identify minimum costs

$$C = \min_{\mathcal{T}} C_{\mathcal{T}} \tag{7}$$

or for stochastic link costs

$$\bar{C} = \min_{\mathcal{T}} E\left[C_{\mathcal{T}}\right] \tag{8}$$

and optimal schedules

$$\mathcal{T}^* = \arg\min_{\mathcal{T}} C_{\mathcal{T}} \tag{9}$$

or again for stochastic costs

$$\mathcal{T}^* = \arg\min_{\mathcal{T}} E\left[C_{\mathcal{T}}\right] \tag{10}$$

III. Quantifying Topology Variation Knowledge

We expect that as the packet's knowledge of node positions increases, it can better make use of mobility. At one extreme, the packet plans its actions with complete knowledge of present and future node positions (section-III-A). In the other extreme a packet might be given current node positions at the beginning of the tour and no further updates (section-III-B). This strategy is similar to *source routing* as used in multi-hop networks [8]. In practice, some level of knowledge between these two extremes is probable, so we also consider a scenario where a packet can obtain updated topology information after each hop, but does not know future topologies (section-III-C).

A. The Omniscient Packet

Suppose that the packet has knowledge of all node itineraries on some interval $[0, T]$. It then has knowledge of all the costs associated with a hop between any two nodes on $[0, T]$ and thus has complete knowledge of the link costs in a graph analogous to that in FIGURE 1. The *omniscient packet* can then identify the cost-minimizing tour using standard dynamic programming methods [9], [10].

We assume that we can select the packet size to control packet transmission time. To simplify the analysis, we choose packet size such that the time required for the transmission and reception of a packet is equal to the step duration δ. Thus, the packet can make only one hop per step. Implicitly, we assume that the channel is stable over such intervals.

Since K is the number of steps and N is the number of nodes, the associated cost graph consists of about NK nodes. With σ the source node and Δ the destination node, we seek minimum cost paths through the graph (also called a *trellis*). To do so, we use dynamic programming [9], [10] whose basic idea is to start at the destination node and work backward finding minimum cost routes recursively.

First, let $K = \frac{T}{\delta}$ and assume that K is an integer larger than 1. Then the cost associated with arrival at the destination is $c_{j\Delta}(T-\delta)$ if the packet is at node j. For a packet at node ℓ at time $T-2\delta$, finding the minimum cost path to the destination requires finding

$$C_{\ell\Delta}(T-2\delta, T) = \min_j \left[c_{\ell j}(T-2\delta) + C_{j\Delta}(T-\delta)\right] \tag{11}$$

and concomitantly, N computations. Before leaving this step, we compute $C_{\ell\Delta}(T-2\delta, T)$ for all $\ell \in \{0, 1, \cdots N-1\}$, an action that requires N^2 computations. Then, finding the path from a node j at time $T-3$ requires us to find

$$C_{j\Delta}(T-3\delta, T) = \min_{\ell} \left[c_{j\ell}(T-3\delta) + C_{\ell\Delta}(T-2\delta, T)\right] \tag{12}$$

which is equivalent to N more computations. Proceeding recursively and making N^2 computations at each step, we can find

$$C_{\sigma\Delta}(0, T) = \min_j \left[c_{\sigma j}(0) + C_{j\Delta}(\delta, T)\right] \tag{13}$$

with approximately KN^2 computations. Normally, we want the packet to reach Δ at $t = T$ or "earlier". Thus, we set $c_{\Delta\Delta}(t) = 0$ for $t \in [0, T]$

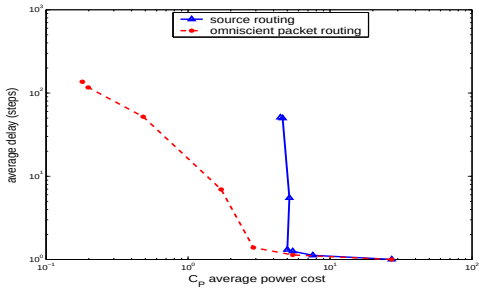

Fig. 2. A comparison of short-sighted and omniscient packet routing for $N = 10$ nodes, $\alpha = 4.0$, $I = 1000$ trials.

We compute average delay-power cost pairs at various (w_d, w_p) values. Since only the relative values of w_d and w_p are important we use a normalized $w_d = 1.0$.

We place N brownian nodes randomly on a rectangular area of $(500 \times 500)m^2$. Many random sets of trajectories are generated and a graph such as FIGURE 1 is formed for each instance. Each time two nodes are chosen at random as the source and destination (σ and Δ) of the packet and the minimum cost from σ to Δ is computed via dynamic programming. Results are then averaged over different instances. We plot average power cost versus delay. These results are used as a benchmark to compare the performances of other strategies which by definition use less information.

B. Short-Sighted Packet

Here, the packet composes the complete tour at $t = 0$ based on expected link costs. The expected cost of a tour $\mathcal{T}$, denoted by a sequence of integers $I_{\mathcal{T}} \equiv \{i_1, i_2, \cdots, i_K\}$, is expressed as:

$$E[C_{\mathcal{T}}|\mathbf{X}_0] = \sum_{k=1}^{K} E[c_{i_{k-1}i_k}(k-1)|\mathbf{X}_0] \tag{14}$$

The expected cost of the optimal tour is:

$$E[C_{\sigma\Delta}(0,T)] = \min_{\mathcal{T}} \sum_{k=1}^{K} E[c_{i_{k-1}i_k}(k-1)|\mathbf{X}_0] \tag{15}$$

and as before, dynamic programming is used to identify an optimal tour. We note that short-sighted packet routing is a simplified form of the source routing used in ad hoc networks.

FIGURE 2 is a comparison between power cost and delay for short-sighted and omniscient packet models. As seen in the figure, the short-sighted packet is incapable of decreasing power cost while sacrificing delay performance. Specifically for large w_p, a short-sighted packet waits longer for more opportune topologies. However, as the time since the departure becomes longer, the difference between the actual cost and the expectation calculated at $t = 0$ increases. Thus, omniscient and short-sighted methods diverge in the low power-high delay region. We note that equation (14) is cumbersome since each expectation of the sum requires an integration on a torus. Thus, from a practical standpoint, finding the optimal tour of short-sighted packets is more complicated than for omniscient packets.

C. The Well-Informed Packet

Suppose the packet has knowledge of the current network topology and the nodal mobility model, but not a detailed node itinerary. How does the packet decide what links to use to reach its destination? There are a variety of approaches to this stochastic problem. Here, we will seek to minimize the mean cost of a tour through the trellis. We assume that tour modifications can be made at each step according to the network topology. $\mathbf{X}(\delta), \mathbf{X}(2\delta), \cdots, \mathbf{X}(T-1)$ are the vectors of random variables corresponding to node positions at different times. Since link costs are now random variables, direct application of dynamic programming is impossible. The solution requires a stochastic dynamic programming approach.

1) Updated Information at Every Step: Let $\mathbf{X}(n\delta)$ be denoted as $\mathbf{X}_n$. At n^{th} epoch, $\mathbf{X}_n$ is known but $\mathbf{X}_{n+1}, \mathbf{X}_{n+2}, \cdots \mathbf{X}_{K-1}$ are not known by the packet. If the nodal mobility is modeled as a stochastic process with stationary and independent increments, like Brownian motion, then $\mathbf{X}$ is a Markov chain and $\mathbf{X}_n$ depends only on $\mathbf{X}_{n-1} = \mathbf{x}_{n-1}$ and $f_{\mathbf{X}_i|\mathbf{X}_{i-1}}(\mathbf{x}_i, \mathbf{x}_{i-1})$. In section-III-C.1 our development closely follows [11], [12].

Let Y_n be the ID of the relay node that carries the packet during step-n. The pair $(\mathbf{x}_n, y_n)$, define the state of the system at step-n. At each step, the packet has N possible control actions to choose from:

(i) it can hop to one of $N-1$ nodes or,
(ii) it can stay put at its current host.

Let A be the set of all possible control actions and $a_n \in A$ be the control action chosen at step-n. The control action a_n depends on $(\mathbf{x}_n, y_n)$:

$$a_n = g_n(\mathbf{x}_n, y_n) \tag{16}$$

The function $g_n()$ is called the *decision rule* for step-n. There are two consequences of an action:

(1) The packet moves to another node in the next step. The relay node at step-$(n+1)$ is:

$$y_{n+1} = h(a_n) = h(g_n(y_n, \mathbf{x}_n)) \tag{17}$$

(2) The packet incurs a cost of $c(y_n, y_{n+1}, \mathbf{x}_n)$, the cost of hopping from node-y_n to node-y_{n+1} when the node positions are $\mathbf{x}_n$. If $y_n = y_{n+1}$, this cost is pure delay cost. Otherwise, it consists of delay and power costs.

Since the cost is additive at each step, the total expected K-step cost from node-i to some destination Δ can be expressed as:

$$C_{i\Delta}(0,K) = \sum_{n=0}^{K-1} c(Y_n, Y_{n+1}, \mathbf{X}_n) \tag{18}$$

where Y_0 and Y_K are deterministic and given by

$$Y_0 = i \quad Y_K = \Delta \tag{19}$$

The goal is to find the sequence of decision rules, $\Pi_{0,K} = (g_0, g_1, \cdots, g_{K-1})$, also called a *policy*, that minimizes the

expected total cost. The problem can be summarized as follows:

$$\min_{\Pi_{0,K}} \left\{ E\left[\sum_{n=0}^{K-1} c(Y_n, Y_{n+1}, \mathbf{X}_n) \right] \right\} \tag{20}$$

such that $Y_{n+1} = h(g_n(Y_n, \mathbf{X}_n))$ with given $Y_0, \mathbf{X}_0, f_{\mathbf{X}_n|\mathbf{X}_{(n-1)}}(\mathbf{x}_n, \mathbf{x}_{n-1})$ and $Y_K = \Delta$.

Let us assume an optimal policy, $\Pi^*_{0,K}$, exists which decides what to do next given the current time index, n, current position vector, $\mathbf{X}_n$. We define $V_{K-n}(i, \mathbf{X}_n)$ as the multihop cost from node i to Δ at step-n and using at most $(K-n)$ of the next steps under $\Pi^*_{0,K}$. The optimal decision rule at this point should be to hop to the node which requires the smallest total expected cost to reach the destination. Thus,

$$V_{K-n-1}(j, \mathbf{X}_{(n+1)}) = \min_i \{c(j, i, \mathbf{X}_{(n)}) + E[V_{(K-n)}(i, \mathbf{X}_n)]\} \tag{21}$$

where the expectation is conditioned on $\mathbf{X}_n$ since at each step the current positions are known and the past positions are irrelevant due to Markovian property of Brownian motion.

By setting initial conditions as:

$$V_0(i, \mathbf{X}_K) = \begin{cases} 0 & i = \Delta \\ \infty & i \neq \Delta \end{cases} \tag{22}$$

and proceeding backward in time, $V_m(y_m, \mathbf{x}_m)$ can be calculated for all $m \in \{1, \ldots, K\}$ and for any $(y_m, \mathbf{x}_m)$. However, there is no explicit expression for the expectations. A numerical solution is intractable for continuous $\mathbf{X}$. Even in the case of a discrete-valued mobility model -like a random walk- computational complexity grows exponentially with the number of steps, K. We also note that to take T to infinity and convert the problem into an infinite horizon dynamic programming does not simplify the problem. So, we seek alternative practical solutions.

2) Infrequent Updates: Lazy Packets: So, we consider another infinite horizon problem where the packet updates its information much less frequently. We call the time between the updates an epoch and denote it by T_E. If T_E is very long (longer than $\tau_D/8$), node positions become approximately IID from epoch to epoch. This assumption facilitates the analysis and leads to a closed form solution. It also represents some practical scenarios: to save power, terminals might be turned off from time to time causing long epochs.

At the beginning of each epoch, the packet *wakes up* and makes a decision. It can either complete its tour to the destination or make some hops without reaching the destination and wait for the next epoch. This sleeping period causes delay, thus it has a cost denoted by C_D where $C_D = w_d T_E$. We will examine different cases where the packet is allowed to take a single hop ($k = 1$) or multiple hops ($k > 1$) during one epoch.

FIGURE 3 illustrates the trellis for the problem. During an epoch, we assume that the link costs will be fixed and equal to their values at the beginning of the epoch for k packet transmission times. Let $u_k(i, j, \mathbf{x}_n)$ be the minimum cost from node-i to node-j when the node position vector is $\mathbf{X}_n = \mathbf{x}_n$

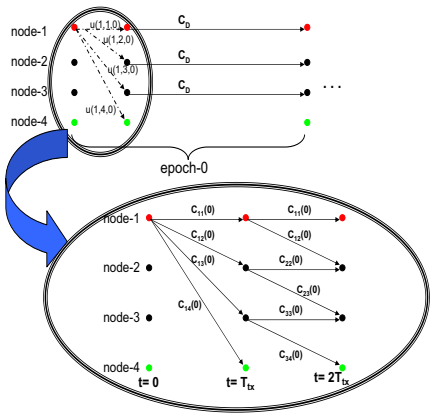

Fig. 3. Illustration of all possible tours for the lazy packet (k = 2) originated at node 1 and destination is node 4 through a network of four nodes. Link costs for the first epoch are shown in detail ($T_{tx} = \delta$). Costs are assumed to be fixed during an epoch.

and the maximum number of hops allowed at each epoch is k. We can write the following recursive equation for $V(i, \mathbf{x}_n)$, the cost from node-i to the destination Δ when the position vector is $\mathbf{x}_n$ and an optimal policy is used.

$$V(i, \mathbf{x}_n) = \min\{\min_{j \neq \Delta}\{u_k(i, j, \mathbf{x}_n) + E[V(j, \mathbf{X}_{(n+1)})]\} + C_D, u(i, \Delta, \mathbf{x}_n)\} \tag{23}$$

where the expectation is with respect to $f_{\mathbf{X}_{(n+1)}}$ as $\mathbf{X}_{(n+1)}$ is independent of $\mathbf{X}_n$. Since any node is the same as any other in the next epoch, $E[V(j, \mathbf{X}_n)]$ is independent of j and will be denoted by $\bar{V}$.

$$V(i, \mathbf{x}_n) = \min\{\min_{j \neq \Delta}\{u_k(i, j, \mathbf{x}_n) + \bar{V}\} + C_D, u_k(i, \Delta, \mathbf{x}_n)\} \tag{24}$$

We note that $u_k(i, j, \mathbf{x}_n)$ is minimized at $j = i$ and its minimum value is equal to zero. We can rewrite equation (24) as:

$$V(i, \mathbf{x}_n) = \min\left\{C_D + \bar{V}, u_k(i, \Delta, \mathbf{x}_n)\right\} \tag{25}$$

Equation (25) shows that the optimal policy for the lazy packet problem is a *threshold rule*. At each epoch, the packet calculates minimum cost to the destination it can take with a maximum of k hops. Then it compares this to the threshold cost $(\bar{V} + C_D)$. If the minimum cost is smaller, then it takes the path immediately. Otherwise, it goes to sleep until the next epoch. We note that the optimal decision rule never allows the packet take some hops in one epoch and then complete the tour in the next one - a result that follows intuition. That is, the packet cannot predict the node positions in the next epoch. If it does not reach the destination in an epoch, in the next epoch all its efforts might have been wasted because it will see an independent and perhaps worse topology.

Moreover, $\bar{V}$ is unique and can be found by solving the

following equation numerically [13]:

$$\begin{aligned}\bar{V} &= C_D\left(\frac{1}{P\{u_k < (\bar{V}+C_D)\}} - 1\right) \\ &+ E[u_k | u_k < (C_D + \bar{V})] \end{aligned} \quad (26)$$

Thus, the optimal threshold, $V_t^* = \bar{V} + C_D$ can be determined for any given pdf for u_k. Unfortunately, obtaining this pdf is practical only for $k = 1$. For $k \geq 2$ statistical simulation methods are more suitable. (For the details see [13].)

IV. Heuristic Approaches

In section-III-C.1, we tried to take advantage of the time correlation of individual node positions. However, the solution of the iterative equations was too complex even for analytically simple Brownian motion. Then, in section-III-C.2, we chose T_E large enough to remove correlation between epochs. Although, this model provided an exact solution, there is usually no need for a packet to sleep that long. So, perhaps a hueristic approach might be benficial.

A. Eager Packets

Here, we assume that T_E is not so long as to guarantee independent topologies from epoch to epoch, but is long enough to allow significant topology change. Specifically, we choose $T_E = \delta$. We use a threshold-based policy like the solution to the problem in section-III-C.2, even though it is not necessarily optimal. As in the *lazy packet* problem, the packet makes its decision based on the current k-hop cost to its destination. If this cost is smaller than some threshold, it moves all the way to its destination. Otherwise, it waits for time T_E. Note that the policy assumes link costs are fixed for k steps and packets use the same most recent topology information for the next k-steps.

At every T_E nodes move and at each node a trellis as in FIGURE 1 is formed. Since the number of hops is limited to k, the trellis consists of N nodes and k steps. As explained in section-III-A, finding the minimum cost on such a trellis requires $0(N^2k)$ computations. The total number of computations depends on when the packet accepts a trellis it observed. Thus, as V_t decreases, the average of the total number of computations increases.

In this strategy there are three parameters: the cost coefficients, (w_d, w_p), and the threshold level, V_t. Thus, unlike the omniscient packet, there is not a single performance curve. Without loss of generality, we set $V_t = 1.0$ and vary the cost coefficients to describe different possible delay-power points.

However, w_d and w_p cannot be chosen arbitrarily. Due to our minimum transmit power constraint, the minimum possible tour between two nodes is a single hop of length d_0. Coefficient pairs used should result a total cost smaller than $V_t = 1.0$ for this minimum cost tour. Otherwise, tour completion is impossible. Let $c(d, w_d, w_p)$ denote the cost of a link of length d when the cost coefficients are w_d and w_p. We recall equation (5) which states that cost of link is in the following form:

$$c(d, w_d, w_p) = w_d c_1 + w_p c_2 d^\alpha \quad (27)$$

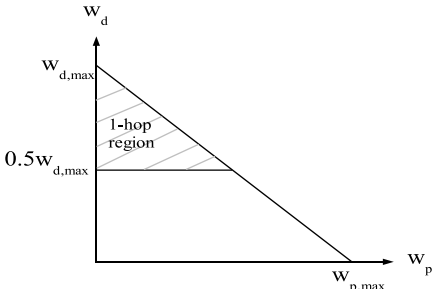

Fig. 4. Shaded area shows the region of valid coefficient pairs. The region where $w_d > 0.5 w_{d,max}$ is called one-hop region.

where c_1 and c_2 are positive numbers. Then, we can calculate the maximum values of w_d and w_p by solving:

$$c(d_0, 0, w_{p,max}) = 1.0 \quad c(d_0, w_{d,max}, 0) = 1.0 \quad (28)$$

We can also find the region of valid cost coefficients. As illustrated in FIGURE 4, this region is a triangle due to the additive cost structure.

We note that it is possible to put a limit on the maximum number of hops the packet takes by choosing an appropriate w_d. For example, when $w_d > w_{d,max}/2$, the packet can never make more than one hop. If the number of hops is limited to one, the strategy becomes a threshold rule on the transmission distance where threshold distance, d_t, corresponding to a particular (w_d, w_p) is obtained by:

$$c(d_t, w_d, w_p) = 1.0 \quad (29)$$

Moreover, in one-hop region the same d_t can be obtained with different coefficient pairs. From equation (27) and equation (29), it can be easily seen that cost coefficients (w_d, w_p) that correspond to the same d_t form a line defined by:

$$c_1 w_d + c_2 d_t^\alpha w_p - 1 = 0 \quad (30)$$

B. Eager Packet Performance

For a performance comparison we evaluate average delay and power cost of our heuristic by simulation for a given w_d and a wide range of w_p. We do not impose a maximum number of hops, k, directly, i.e. we set $k = \infty$, and exercise control only through the cost coefficients.

In FIGURE 5 we compare omniscient packets with eager packets that use different w_d's. On each curve for eager packet w_d is kept fixed and w_p is varied. As expected, performance of the omniscient packet is better than the eager packet. However, unlike the short-sighted packet, the eager packet follows the same trend as the omniscient packet while performing within approximately a factor of five.

We see that in the low power-high delay region (delay$>$ 1000 steps), the performance of the eager packet is insensitive to w_d. On the other hand, in the high power-low delay region (delay$\leq$ 1000 steps), performance depends on w_d. Smaller w_d seems to reduce the delay in this region – a seemingly counterintuitive result.

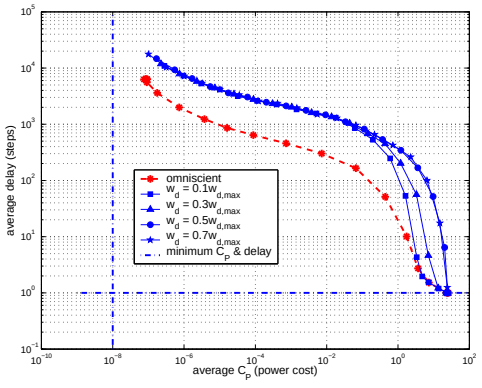

Fig. 5. A comparison of the eager packet and the omniscient packet. $N = 15$ nodes, propagation constant, α is equal to 4.0

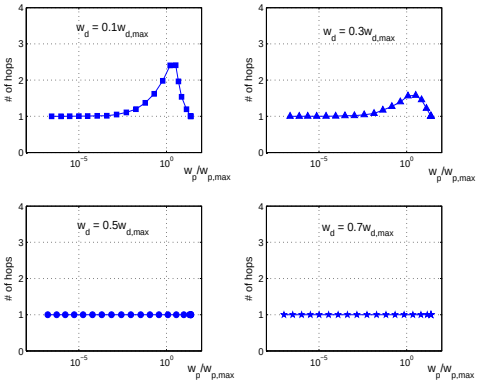

Fig. 6. Average number of hops at different cost coefficients. $N = 15$ nodes.

To better understand this seeming paradox, in FIGURE 6 we plot the average number of hops as a function of w_d and w_p. The number of hops increases as we decrease w_d and, as explained previously, reduces to 1 when $w_d \geq 0.5w_{d,max}$. It is observed that even for the cases where $w_d < 0.5w_{d,max}$, the packet takes approximately one hop at very high and very low w_p. In the first case, very high w_p, the packet takes one hop because most low-power tours consist of a single hop. In other words, statistically it is very unlikely to observe a topology where the minimum cost tour from the source to the destination has more than one hop. In the very low w_p case, all hops have almost the same cost. Then, there is no motivation to take multiple hops instead of a direct transmission from source to destination. When we re-examine FIGURE 5 in light of FIGURE 6, we see that smaller w_d can achieve shorter delays when there is a potential benefit from multiple hops. In this experiment, making multiple hops is advantageous in the region where average delay is $1 - 500$ steps.

In a fixed network, it is known that allowing multiple hops can save power. The region where the delay is close to 1 can be interpreted as a fixed network since the topology does not change during the packet delivery. The only difference is at the last portion of the curve where the delay cost is dominant and our policy delivers the packet in the smallest possible time, i.e, making a single hop directly to the destination.

V. Summary

In this paper, we examined the power-delay tradeoff for a packet that receives negligible amount of interference from the network. We analyzed optimal packet decisions based on our cost structure under three different levels of position knowledge: complete trajectory knowledge (omniscient), initial topology knowledge (short-sighted) and topology update after each hop (well-informed). Since omniscient packet routing has complete topology information at the start of a tour, it is used as a performance benchmark for policies which use less information. We have seen that the short-sighted packet is ineffective in the lower/higher cost/delay region.

Since the solution for the well-informed packet was too complex, we made a set of assumptions (lazy packet model) that led to a simple solution – a threshold rule based on the current network topology. Then we relaxed the assumption of independent position from step to step required for solution of the lazy packet policy, but kept the idea of a threshold (eager packet). We found that the eager packet policy followed the same trends as omniscient packet policies while achieving a delay performance within approximately a factor of 5.

In closing, we note that from a practical and computational perspective, only the eager packet strategy is simple enough to be used in network studies in which packets interact. Thus, we make use of the eager packet threshold method in a network setting to derive policies which trade off average delay versus throughput in a companion paper [7].

References

[1] D. Goodman, J. Borras, N. Mandayam, and R. Yates. Infostations: A New System Model for Data and Messaging Services. In *IEEE VTC*, volume 2, pages 969–973, 1997.

[2] R. H. Frenkiel, B. R. Badrinath, J. Borras, and R. Yates. The Infostations Challenge: Balancing Cost and Ubiquity in Delivering Wireless Data. *IEEE Personal Communications*, 7(2):66–71, April 2000.

[3] M. Grossglauser and D. Tse. Mobility Increases the Capacity of Ad Hoc Networks. *IEEE/ACM Transactions on Networking*, 10(4):447–486, Aug. 2002.

[4] S. Toumpis and A. Goldsmith. Capacity Regions for Wireless Ad Hoc Networks, 2003. To appear in the IEEE Transactions on Wireless Communications.

[5] T. A. ElBatt, S. V. Krishnamurthy, D. Connors, and S. K. Dao. Power Management for Throughput Enhancement in Wireless Ad-Hoc Networks. In *ICC (3)*, pages 1506–1513, 2000.

[6] L. Kleinrock and J. Silvester. Spatial Reuse in Mutlihop Packet Radio Networks. In *Proc. of IEEE*, volume 75, pages 156–167, Jan. 1987.

[7] F. Atay and C. Rose. Exploiting Mobility in Mobile Infostation Networks to Maximize Throughput. Submitted to ISART2004.

[8] David B Johnson and David A Maltz. Dynamic source routing in ad hoc wireless networks. In Imielinski and Korth, editors, *Mobile Computing*, volume 353. Kluwer Academic Publishers, 1996.

[9] D. Bertsekas. *Dynamic Programming: Deterministic and Stochastic Models*. Prentice-Hall, 1987.

[10] R. G. Gallager. *Discrete Stochastic Processes*. Kluwer, 1996.

[11] R. E. Stokey, R. E. Lucas, and E. C. Prescott. *Recursive Methods in Economic Dynamics*. Harvard University Press, 1989.

[12] G. Violante. Notes on Discrete Time Stochastic Dynamic Programming. Available at http://www.econ.iastate.edu/tesfatsi/dpbasic.violante.pdf.

[13] F. Atay. Exploiting Mobility in Mobile Ad Hoc Networks. Master Thesis, Dept. of Electrical and Computer Engineering, Rutgers University, Piscataway, NJ, USA, 2003.