Latency and Connectivity Analysis Tools for Wireless Mesh Networks

Phoebus Wei-Chih Chen
S. Shankar Sastry

Electrical Engineering and Computer Sciences
University of California at Berkeley

Technical Report No. UCB/EECS-2007-87
http://www.eecs.berkeley.edu/Pubs/TechRpts/2007/EECS-2007-87.html

June 29, 2007

Acknowledgement

The authors would like to thank Songhwai Oh, Ian Tan, Kris Pister, and David Tse for offering feedback on the ideas in this paper, and particularly Kris Pister for providing more details about TSMP and 802.15.4 radios.

Latency and Connectivity Analysis Tools for Wireless Mesh Networks

Phoebus Chen and Shankar Sastry
Department of Electrical Engineering and Computer Sciences
University of California, Berkeley
Berkeley, California 94720
{phoebusc,sastry}@eecs.berkeley.edu

***Abstract*—There has been a recent rise in interest in building networked control systems over a wireless network, whether they be for robot navigation, multi-robot systems, or traditional industrial automation. The wireless networks in these systems must deliver packets between the controller and the actuators/sensors reliably and with low latency. Furthermore, they should be amenable to modeling and characterization so they can be designed as part of a complete control system. Mesh networks are particularly suited for control applications because they provide greater reliability through path diversity.**

This paper introduces tools for characterizing the end-to-end connectivity of two points in a wireless mesh network as a function of latency. In particular, we use tools derived from Markov chain models to compare end-to-end connectivity in two routing protocols running on the Data Link/MAC layer provided by Dust Network's Time Synchronized Mesh Protocol (TSMP): Directed Staged Flooding (DSF) and Dust Network's Unicast Path Diversity (UPD). These models also allow us to calculate the traffic load, the sensitivity of end-to-end connectivity to link estimation error, and the robustness of the network to node failure. The paper gives an example of how these tools can be used to evaluate the feasibility of running control applications over sensor networks.

Latency and Connectivity Analysis Tools for Wireless Mesh Networks

I. Introduction

Wireless mesh networking has enabled a new generation of pervasive devices with the potential to provide reliable communication in environments with limited fixed infrastructure. Wireless sensor and actuator networks, sometimes simply referred to as sensor networks, are one such class of devices which can use mesh networking to connect sensors and actuators that monitor our environment and control other instruments [1]. Sensor networks enable a large variety of applications including outdoor environmental monitoring for scientific research, diagnosing civil structures for damage under earthquakes, monitoring the sick and elderly for assisted living at home, assisting firefighters navigating through a burning building, providing situational awareness to soldiers on a battlefield, mapping out terrain and identifying moving objects for robot navigation and surveillance, and sensing and control of industrial automation equipment, among others. Feedback control systems are among the hardest types of applications for sensor networks because they place stringent requirements on reliability and latency. These applications motivate the need for tools to characterize wireless mesh networking on sensor networks for control systems.

A. Control over Lossy Networks

The recent increased interest in using wireless networks for industrial automation culminated in the formation of the ISA-SP100 committee to set up an industrial wireless standard [2] and the adaptation of the wired process automation protocol HART to a wireless protocol, WirelessHART [3]. The current version of both the ISA-SP100 and WirelessHART standards plan to build on the PHY layer provided by the IEEE 802.15.4 standard [4] for low-power, ad-hoc, wireless, personal area networks.

The key issues in using wireless communications for control systems is reliability and latency. The designer of the control system needs to know the probability of end-to-end delivery of the packet, p_{net}, as a function of delay, t_d, to provide performance guarantees on the controller. Many papers in the area of *Networked Control Systems* study the impact of packet loss on the stability of discrete-time estimators [5], [6], [7], assuming that packets arriving after a deadline (the sampling period) are lost. But despite using the moniker "network", the results of these papers are usually derived using a simple, point-to-point communication channel with fixed delay. For instance, in [6], Sinopoli et al. assume the packet loss in the channel can be modeled by an i.i.d. Bernoulli random variable.

The goal of this paper is to model examples from two classes of TDMA mesh networks for control systems, multi-path routing with retransmissions and constrained flooding. We wish to derive the function $p_{net}^{(t_d)}$ relating the probability of end-to-end delivery to delay for a packet in a wireless mesh network providing communication for a control system so we can use the existing theory in Networked Control Systems to characterize the system's stability and performance. The paper focuses on TDMA networks because of the difficulty modeling and providing probabilistic guarantees on latency for networks using CSMA/CA contention protocols. It focuses on mesh networks because multiple paths between a source and destination are necessary for good end-to-end reliability.

B. Related Work on Multi-path Routing

Many of the standard routing protocols implemented in TinyOS [8], [9], an open source operating system for sensor networks, are single-path, many-to-one routing algorithms for collecting data from the network. MultiHopRouter, MultiHopLQI [10], MintRoute [11], and Drip and Drain [12] are all variants on minimum weight path routing, where the weights are some function of the link quality estimates and hop count.

To increase reliable end-to-end delivery of packets, many routing schemes propose sending multiple copies of a packet on multiple paths. These range from controlled, probabilistic flooding schemes like ARRIVE [13] to schemes that code the data over a set of packets and send them along disjoint or braided (partially disjoint) paths such that only a subset of the packets need to be received for reconstruction [14]. Multi-path routing schemes are also distinguished by whether an end-to-end path is selected at the source for a packet, such as the braided and disjoint paths of [15], or whether the packet can switch paths during transit, as in "true mesh" routing protocols like Unicast Path Diversity[1] on TSMP [16] and ARRIVE. The protocols studied in this paper will be of the latter type.

Many of the multi-path routing papers use simulations to demonstrate qualitative features of their routing schemes. For instance, in [15], Ganesan et al. propose using the gradient-based route discovery mechanism in Directed Diffusion [17] to establish disjoint and braided paths between the source and destination. Then, the paper uses simulations to evaluate how patterned (geographically clustered) failures and isolated (uniformly distributed) failures affect the resilience (probability at least one alternate path is available given that at least one node on the primary path has failed) of the network. Similarly, [13] uses simulations to evaluate the resilience and energy/reliability tradeoff of ARRIVE.

A small set of papers try to mathematically model and analyze the benefits of multi-path routing, but they either model at the level of paths or assume networks with a very

[1]The name *Unicast Path Diversity* is not explicitly mentioned in the reference, but this is the name of the routing protocol that is described.

large number of nodes. In [14], Dulman et al. perform some simple analysis to get the tradeoff between traffic and reliability, but the analysis does not consider latency. Furthermore, the calculations use the end-to-end connection probability of disjoint *paths*, not individual link probabilities, and hence do not account for varying path lengths or link probabilities. In [18], Nasipuri et al. propose a multi-path extension to DSR and the analysis focuses on finding the statistics of the time between successive route discoveries. Again, the paper builds on a path model with path lifetimes drawn from a distribution instead of a link model with individual link probabilities. In [19], the authors use a geometry-based argument on networks with a very large number of nodes to argue that k-shortest path routing algorithms only distribute the load evenly through a network when it uses a very large number of paths.

This paper takes a different approach from the papers mentioned above, deriving link-probability-based analysis tools applicable to networks of any size for two examples of mesh routing protocols. In Sections II and III, we present the models and analysis tools for Unicast Path Diversity and Directed Staged Flooding, the two mesh routing protocols. This is followed by a comparison of these two protocols in the context of control systems in Section IV, and finally a discussion on directions for future work in Section V.

II. Unicast Path Diversity

Dust Networks, Inc. proposed Unicast Path Diversity (UPD) over Time Synchronized Mesh Protocol (TSMP) [16] for reliable networking in sensor networks. UPD was designed for industrial automation, building automation, and security and defense applications, where one may wish to close a control loop around a sensor network. The algorithm exploits frequency, time, and space diversity to achieve what they claim is over 99.9% typical network reliability [20]. We use a general Mesh TDMA Markov Chain (MTMC) model to analyze the performance of UPD for incorporation into a control system.

A. Modeling Characteristics

UPD over TSMP (hereafter referred to simply as UPD) is a network and MAC protocol that has several defining characteristics [16]:

1) Mesh/Multi-Path Routing
2) Time Synchronized Communication
3) Frequency Hopping
4) Automatic Node Joining/Network Formation
5) Secure Message Transfer

This paper is not concerned with the authentication, encryption, and integrity check security mechanisms for packets, and it defers modeling automatic network and routing schedule formation for future work. As such, we model UPD as a frequency-hopping TDMA scheme with multi-path routing.

UPD forms multiple, interleaved routing paths from many nodes to one sink node (the network manager/base-station node). That is, each node has multiple parents and the routing graph has no cycles. The links selected for routing are bidirectional, and hence every transmission on a link can be acknowledged. If a packet transmission is not acknowledged, it is queued in the node for retransmission.

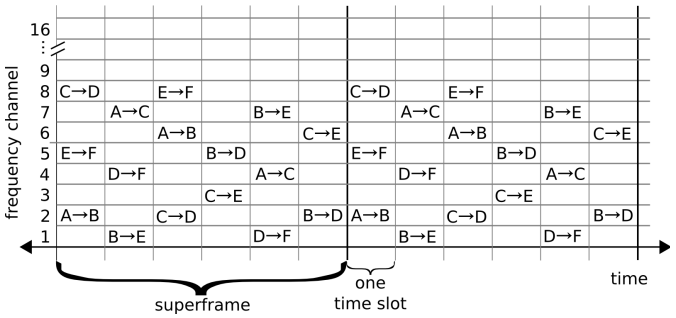

Fig. 1. Example of a UPD schedule with superframes and time slots. Here, only 8 of the 16 frequency channels are used.

UPD uses time synchronization between the nodes so the nodes can follow a TDMA routing schedule, ensuring that there are no packet collisions as in CSMA MAC protocols. Time is divided into time slots, and grouped into superframes (See Figure 1). At each time slot, pairs of nodes are scheduled for transmitting a packet on different frequencies. The superframe containing the schedule of transmissions is repeated over time.

At different time slots within a superframe, a given pair of nodes will try to communicate on different frequencies (frequency hopping). While there is significant work in selecting orthogonal frequency hopping schedules, such as Latin squares [21], our model only uses frequency hopping to justify the assumption that links are independent over retransmissions.

As mentioned in Section I-A, given these characteristics of a mesh TDMA routing scheme, we wish to know the probability of end-to-end connectivity, p_{net}, as a function of delay, t_d. In addition, we wish to characterize the robustness of the network to node removal and perturbation of link probabilities. Robustness to node removal involves finding the traffic distribution over the nodes in the network — i.e. identifying *hot spots* in the network. Identifying hot spots gives us a sense of how vulnerable the network is to the compromise of an individual node, while knowledge of the traffic distribution together with a model of node energy consumption allows us to compute the lifetime of the network. Robustness to link probability perturbation gives a sense of the reliability of our analysis despite errors in link probability estimation.

To construct our model of mesh TDMA routing, we assume knowledge of the routing schedule, the routing topology (which can be derived from the routing schedule), and all the link probabilities. Furthermore, we study single packet transmission in the network and do not analyze the effects of queuing. Further discussion on the implications of these modeling assumptions can be found in the Section IV-D.

B. Mesh TDMA Markov Chain Model

Let us represent the routing topology as a graph $G = (\mathcal{V}, \mathcal{E})$, and denote a node in the network as $i \in \mathcal{V} = 1, \ldots, N$, and a link in the network as $l \in \mathcal{E} \subset \{(i,j) \mid i,j \in \mathcal{V}\}$, where

$l = (i, j)$ represents a link transmitting from node i to node j. Time t will be measured in units of time slots, and let T denote the number of time slots in a superframe. The link success probability for link $l = (i, j)$ at time slot t is denoted $p_l^{(t)}$, or $p_{ij}^{(t)}$. We set $p_l^{(t)} = 0$ when link l is not scheduled to transmit at time t. Note that this allows for different link probabilities when transmitting between a pair of nodes at different frequencies on different time slots.

For a packet originating from a source node a routed to a sink node b, we wish to compute $p_{net}^{(t_d)}$, the probability the packet reaches b at or before time t_d has elapsed. A derivation for the special case of routing along a single path with retransmissions is given in Appendix A.

We can calculate $p_{net}^{(t_d)}$ by noticing a Markov property of the packet transmissions. Let $l = (i, j) \in \mathcal{E}$, $S_l^{(t)}$ denote the event that a packet is at node i and is successfully transmitted on link l at time t, and $\bar{S}_l^{(t)}$ denote the event that a packet is at node i but link l fails at time t. Then for all $t_0 < t_1$, $k \in \mathcal{E}$,

$$S_k^{(t_0)} \perp\!\!\!\perp S_l^{(t_1)} \mid \text{packet at } i \text{ between } t_1 - 1 \text{ and } t_1 \quad . \tag{1}$$

So if $k_1, k_2, \ldots, k_{t-1}$ are links along a path with $k_{t-1} = (h, i)$ for some node h, then

$$\mathbb{P}(S_l^{(t)} | S_{k_{t-1}}^{(t-1)}, S_{k_{t-2}}^{(t-2)}, \ldots, S_{k_1}^{(1)}) = \mathbb{P}(S_l^{(t)} | S_{k_{t-1}}^{(t-1)}) \quad . \tag{2}$$

Let us construct a time-varying, discrete-time Markov chain to compute $p_{net}^{(t_d)}$ for routing over TDMA mesh topologies.

Mesh TDMA Markov Chain Model Let the set of states in the Markov chain be the nodes in the network, $\mathcal{V}$. The transition probability from state i to state j at time t is simply $p_{ij}^{(t)}$, with $p_{ii}^{(t)} = 1 - \sum_{j \neq i} p_{ij}^{(t)}$. Let $P^{(t)} = [p_{ij}^{(t)}]^T \in [0, 1]^{N \times N}$ be the transition probability matrix for a time slot and $P^{(\underline{T})} = P^{(T)} P^{(T-1)} \ldots P^{(1)}$ be the transition probability matrix for a repeating superframe.[2] Assume

$$P^{(T+h)} = P^{(cT+h)}, \qquad \forall c, h \in \mathbb{Z}_+ \tag{3}$$

meaning that the link probabilities in a time slot do not vary over superframes.

A packet originating at node a is represented by $\mathbf{p}^{(0)} = \mathbf{e}^{[a]}$, where $\mathbf{e}^{[a]}$ is an elementary vector with the a-th element equal to 1 and all other elements equal to 0. Then,

$$\mathbf{p}^{(t_d)} = P^{(t_d)} \ldots P^{(2T+1)} \underbrace{P^{(2T)} P^{(2T-1)} \ldots P^{(T+1)}}_{P^{(\underline{T})}} \underbrace{P^{(T)} P^{(T-1)} \ldots P^{(1)}}_{P^{(\underline{T})}} \mathbf{p}^{(0)} \tag{4}$$

represents the probability distribution of the packet over the nodes at time t_d. ■

The sink node b is an absorbing state in the Markov chain, meaning there are no transitions out of that state (in our routing schedule, a packet is never transmitted from the sink to another node in the network). This means $p_{net}^{(t_d)} = \mathbf{p}_b^{(t_d)}$ (the b-th element of the vector $\mathbf{p}^{(t_d)}$), where $\mathbf{p}_b^{(t_d)}$ is the probability that the packet sent at time 1 reaches the sink by time t_d. A good routing schedule would have $p_{net}^{(t_d)} \xrightarrow{t_d \to \infty} 1$, meaning the packet will eventually reach the sink. This condition is satisfied when the MTMC model has only one recurrent class consisting of the sink (See [22] for a discussion on recurrent classes in Markov chains).

[2] $[0, 1]$ denotes the closed interval between 0 and 1.

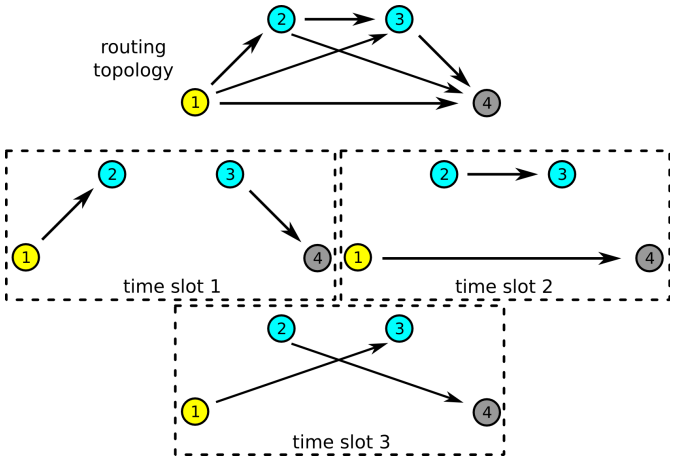

Fig. 2. Multi-path routing example corresponding to Equation 5.

C. MTMC Examples and Discussion

An example of a small UPD routing schedule is given in Figure 2, where p_{ij} is the link probability for link (i, j) and $\bar{p}_{ij} = 1 - p_{ij}$. In this example, the transmission schedule was selected such that a node does not listen and transmit in the same time slot. We get the transition probability matrices,

$$P^{(1)} = \begin{bmatrix} \bar{p}_{12} & 0 & 0 & 0 \\ p_{12} & 1 & 0 & 0 \\ 0 & 0 & \bar{p}_{34} & 0 \\ 0 & 0 & p_{34} & 1 \end{bmatrix} \quad P^{(2)} = \begin{bmatrix} \bar{p}_{14} & 0 & 0 & 0 \\ 0 & \bar{p}_{23} & 0 & 0 \\ 0 & p_{23} & 1 & 0 \\ p_{14} & 0 & 0 & 1 \end{bmatrix}$$

$$P^{(3)} = \begin{bmatrix} \bar{p}_{13} & 0 & 0 & 0 \\ 0 & \bar{p}_{24} & 0 & 0 \\ p_{13} & 0 & 1 & 0 \\ 0 & p_{24} & 0 & 1 \end{bmatrix}$$

$$\mathbf{p}^{(0)} = \begin{bmatrix} 1 & 0 & 0 & 0 \end{bmatrix}^T \qquad P^{(\underline{3})} = P^{(3)} P^{(2)} P^{(1)} \tag{5}$$

The MTMC model is flexible enough to represent routing topologies and schedules not used by UPD. For instance, UPD avoids creating cycles in the routing graph, as one would want from a good routing algorithm. The MTMC model, however, can model routing cycles that may arise when the network malfunctions. We can still calculate $p_{net}^{(t_d)}$, and we still have $p_{net}^{(t_d)} \xrightarrow{t_d \to \infty} 1$ if no recurrent classes besides the sink are added to the Markov chain. The MTMC model can also be extended to represent mesh networks with multiple collection points (ex. two internet gateways to a sensor network). In this case, if we let $\mathcal{B}$ be the set of sinks, $p_{net}^{(t_d)} = \sum_{i \in \mathcal{B}} \mathbf{p}_i^{(t_d)}$. If we wish to model a network sending packets individually addressed to different sink nodes, we would use a separate MTMC model for each sink in the network. Of course, this still assumes only one packet is in the network at any point in time, since we do not model queuing. Finally, if we wish to model schedules that never retransmit packets, we simply remove

the requirement in Definition II-B that $p_{ii}^{(t)} = 1 - \sum_{j \neq i} p_{ij}^{(t)}$, instead replacing it with $p_{ii}^{(t)} = 0$. To ensure that the transition probability matrix $P^{(t)}$ is a column stochastic matrix, we add a dummy state $N+1$ to represent a packet being lost after transmission. Now, $P^{(t)} = [p_{ij}^{(t)}]^T \in [0,1]^{N+1 \times N+1}$, where $p_{i(N+1)}^{(t)} = 1 - \sum_{j \neq i} p_{ij}^{(t)}$, $p_{(N+1)i}^{(t)} = 0$ for all $i \neq N+1$, and $p_{(N+1)(N+1)}^{(t)} = 1$.

Also, recall that the definition of the MTMC model assumes the link probabilities in a superframe do not vary with t. This assumption is valid if the link probabilities vary on a slower time scale than the time to transmit a packet. This means that if we only look at $\mathbf{p}^{(cT)}, c \in \mathbb{Z}_+$, the Markov chain at this time scale is not time-varying. To accommodate schedules and link probabilities that vary over time, we simply remove the restriction imposed by Equation 3. This, however, means that $\mathbf{p}^{(cT)} \neq (P^{(\underline{T})})^c \mathbf{p}^{(0)}$ and we cannot directly use some of the tools presented in Section II-D that depend on time-invariance of the Markov chain.

D. MTMC Analysis

1) Network-wide Rate of Convergence for $p_{net}^{(t_d)}$: In the previous section, we showed how to use the MTMC model to calculate $p_{net}^{(t_d)}$ for a packet transmitted from each node in the network to the sink. We can also get the rate of convergence of $p_{net}^{(t_d)}$ to 1 for the *entire* network from $P^{(\underline{T})}$. This may be a useful metric for designing routing schedules to optimize the performance of the network.

If we renormalize time in units of T and look at the network after each superframe of transmission, we claim that the rate of convergence of $\mathbf{p}^{(t)}$ as $t \to \infty$, regardless of $\mathbf{p}^{(0)}$, is exponential with rate parameter ρ_*, the magnitude of the largest eigenvalue of $P^{(\underline{T})}$ with absolute value strictly less than 1. Thus, ρ_* gives a sense of how the *worse case* end-to-end connection probability in the network varies as a function of delay. The key requirement is that we have a good routing schedule where $p_{net}^{(t_d)} \xrightarrow{t_d \to \infty} 1$, meaning any packet will eventually reach the sink. This is stated more precisely in the following theorem, which is proved in Appendix B.

Theorem 2.1 (MTMC $p_{net}^{(t_d)}$ converges exponentially to 1): Let $P^{(\underline{T})} \in [0,1]^{N \times N}$ be a column stochastic matrix with $\lim_{k \to \infty} (P^{(\underline{T})})^k \mathbf{p} = \mathbf{e}^{[b]}$ for all probability vectors $\mathbf{p}$. Here, $\mathbf{e}^{[b]}$ is an elementary vector with the b-th element equal to 1 and all other elements equal to 0, meaning that the routing topology has a unique sink node b which is the unique recurrent state in the Markov chain. Then,

$$p_{net}^{(t_d)} \geq 1 - Ck^{J-1}(\rho_*)^{k-J+1}, \quad k = \left\lfloor \frac{t_d}{T} \right\rfloor \tag{6}$$

for some constant C dependent on the initial distribution $\mathbf{p}^{(0)}$, $J \in \mathbb{Z}_+$ the size of the largest Jordan block of $P^{(\underline{T})}$, and $\rho_* = \max\{|\lambda| : \lambda \text{ is an eigenvalue of } P^{(\underline{T})} \text{ and } |\lambda| < 1\}$. ■

In the special case where $P^{(\underline{T})}$ is a diagonalizable matrix, we have $J = 1$ and Equation 6 becomes

$$p_{net}^{(t_d)} \geq 1 - C(\rho_*)^k, \quad k = \left\lfloor \frac{t_d}{T} \right\rfloor \quad . \tag{7}$$

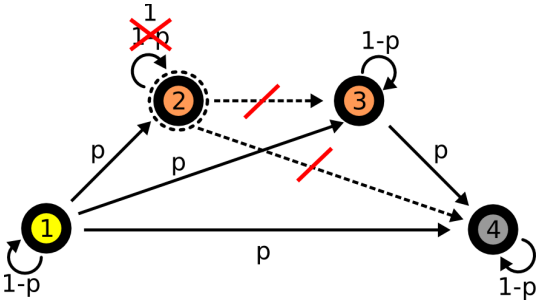

Fig. 3. Illustration of how to create absorbing states in the Markov chain to calculate the probability that a packet sent from node 1 to node 4 passes through node 2 by time t, using the routing topology of Figure 2.

2) Traffic Distribution: In the MTMC model, $\mathbf{p}^{(t)}$ represents the probability distribution of the packet *at* time t. To identify hot spots in the network, it is more useful to compute the probability that the packet visits a node i *at or before* time t, $\tilde{\mathbf{p}}_i^{(t)}$. This can be done by making i an absorbing state in the MTMC model and finding $\mathbf{p}_i^{(t)}$ on the new model.

In other words, $\forall t \in \mathbb{N}, \forall j \in \mathcal{V}$, let

$$\begin{aligned} \tilde{P}_{ji}^{(t)} &= 0 \\ \tilde{P}_{ii}^{(t)} &= 1 \\ \tilde{P}_{mn}^{(t)} &= P_{mn}^{(t)} \quad \forall m, n \in \mathcal{V}, n \neq i \end{aligned}$$

(See Figure 3). The resulting model has two absorbing states, b and i. $\alpha_i^{(t)} = \tilde{\mathbf{p}}_i^{(t)} = \tilde{P}^{(t)} \tilde{P}^{(t-1)} \ldots \tilde{P}^{(1)} \mathbf{p}^{(0)}$ is the probability that the packet visits node i in the original model at or before time t, while $\alpha_b^{(t)} = \tilde{\mathbf{p}}_b^{(t)}$ is the probability that the packet arrives at the sink through an alternate path disjoint with node i.[3]

To find $\alpha_i = \lim_{t \to \infty} \tilde{\mathbf{p}}_i^{(t)}$, the probability the packet ever visits node i, we solve a system of equations for the probability that any state $j \neq i$ is absorbed into state i, as mentioned in [22].

Theorem 2.2 (Absorption Probability Equations [22]): For a given Markov chain, choose an absorbing state i. Then, the probabilities α_j of reaching state i starting from j are the unique solution to the equations

$$\begin{aligned} \alpha_i &= 1 \\ \alpha_j &= 0 \quad \text{for all absorbing } j \neq i \\ \alpha_j &= \sum_{k=1}^{N} p_{jk} \alpha_k \quad \text{for all transient } j \end{aligned} \tag{8}$$

■

3) Link Perturbation: We would like to know the sensitivity of $p_{net}^{(t_d)}$ to errors in link probability estimation. The way that $p_{net}^{(t_d)}$ varies with link probability p_{ij} is not always obvious. For instance, there are situations where increasing the probability of a link in the routing topology results in a *decrease* in $p_{net}^{(t_d)}$, as illustrated in the example of Figure 4.

In fact, we *cannot* get a bound on the range of $p_{net}^{(t_d)}$ by simply recomputing $p_{net}^{(t_d)}$ using the endpoints on the range of link probabilities $p_{ij}^{(t)} + \epsilon$ and $p_{ij}^{(t)} - \epsilon$. To see this, let the

[3]The extra notation using α is for consistency with DSFMC and will be used later in Section IV.

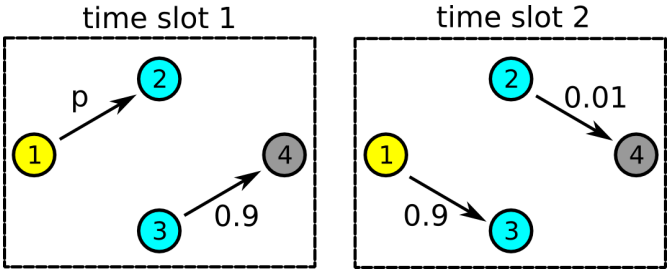

Fig. 4. Example where increasing the link probability p actually results in *lower* $p_{net}^{(t_d)}$ for all t_d.

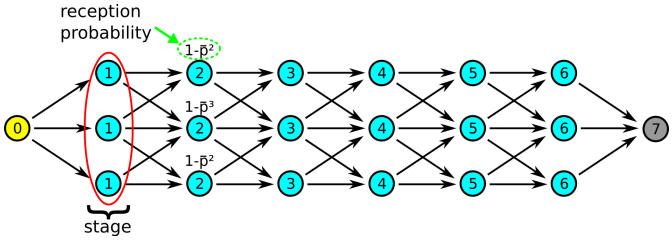

Fig. 5. Directed Staged Flooding example on a wide path topology containing stages with a path width of 3. Discussed in more detail in Section III-B.

actual link probability of a link at time slot t be $\hat{p}_{ij}^{(t)} = p_{ij}^{(t)} + \delta$, where δ is unknown to the user but the user knows that $\hat{p}_{ij}^{(t)}$ lies within $\pm\epsilon$ of the estimate $p_{ij}^{(t)}$. We can write the actual end-to-end transition probability matrix as

$$\begin{aligned} \hat{P}^{(\underline{T})} &= P^{(T)} \dots P^{(t+1)}(P^{(t)} + \delta E^{(t)})P^{(t-1)} \dots P^{(1)} \\ &= P^{(\underline{T})} + \delta \underbrace{P^{(T)} \dots P^{(t+1)} E^{(t)} P^{(t-1)} \dots P^{(1)}}_{F} \end{aligned} \quad (9)$$

where $E^{(t)}$ is a matrix with -1 at $E_{ii}^{(t)}$, 1 at $E_{ji}^{(t)}$, and 0s elsewhere. Here, for simplicity, we assumed that a link is used only once in a superframe. Define $\hat{\mathbf{p}}^{(t)}$ as the actual probability distribution at time t. Then,

$$\begin{aligned} \hat{\mathbf{p}}^{(T)} - \mathbf{p}^{(T)} &= (\hat{P}^{(\underline{T})} - P^{(\underline{T})})\mathbf{p}^{(0)} \\ &= \delta F \mathbf{p}^{(0)} \end{aligned} \quad (10)$$

and

$$\begin{aligned} \hat{\mathbf{p}}^{(2T)} - \mathbf{p}^{(2T)} &= (\hat{P}^{(\underline{2T})} - P^{(\underline{2T})})\mathbf{p}^{(0)} \\ &= (\delta(P^{(\underline{T})}F + FP^{(\underline{T})}) + \delta^2 F^2)\mathbf{p}^{(0)} \end{aligned} \quad (11)$$

Note that in Equation 11, δ enters into the equation quadratically. Thus, because of retransmissions on links (manifested by repeating superframes), it is not clear that $p_{net}^{(t_d)}$ would vary monotonically with the perturbation of the link. Thus, we cannot use $p_{ij}^{(t)} \pm \epsilon$ to bound $p_{net}^{(t_d)}$.

The alternative is to try bounding the distance of the eigenvalues $\hat{\lambda}$ of $\hat{P}^{(\underline{T})}$ from the eigenvalues λ of $P^{(\underline{T})}$, a standard problem in matrix perturbation analysis. In other words, if $\hat{\lambda}_x$ is an eigenvalue of $\hat{P}^{(\underline{T})} = P^{(\underline{T})} + \delta F, \delta \in (-\epsilon, +\epsilon)$, then there is some eigenvalue λ_y of $P^{(\underline{T})}$ such that $|\hat{\lambda}_x - \lambda_y| < \mathcal{C}(F, \epsilon)$, where $\mathcal{C}(\cdot,\cdot)$ is some function of F and ϵ. There are several standard techniques to do this, some that require $P^{(\underline{T})}$ to be diagonalizable or $P^{(\underline{T})}$ to be normal ($A^*A = AA^*$), which may not always hold. These techniques are applicable on a case by case basis. For more details, see [23]. Note that the problem becomes more complicated if we consider the estimation error of multiple links. Equation 9 will need to be modified to incorporate multiple $E^{(t)}$, which may in turn result in a large perturbation matrix F and a loose bound on the $\hat{\lambda}_x$.

III. Directed Staged Flooding

To increase the reliability of multi-path routing on wireless networks without increasing latency, it seems natural to try to exploit the broadcast nature of the medium to transmit multiple copies of a packet simultaneously in one transmission. At one extreme, we flood the network and waste a lot of bandwidth if each node always transmits to all its neighbors. But what if a node multicasts a packet to a subset of its neighbors? In effect, a packet will try multiple links in one transmission instead of trying each link sequentially, potentially providing better end-to-end connectivity with less latency. Here, instead of retransmitting a packet after knowledge that the link failed we are effectively "*preemptively* retransmitting" the packet on multiple links.

We propose a simple constrained flooding scheme called *Directed Staged Flooding* (DSF) for one-to-many and one-to-one routing, focusing on the latter. We use a Directed Staged Flooding Markov Chain (DSFMC) model to find $p_{net}^{(t_d)}$. As with UPD, we build the model assuming we are provided with a routing schedule and all the link probabilities. We leave the development of an algorithm to construct such a routing schedule for future work. The characteristics of DSF are described in the next section.

A. Modeling Characteristics

In DSF we assume that, like UPD, the nodes follow a TDMA routing schedule. During a transmission each node transmits to a subset of its neighboring nodes. Furthermore, we group the nodes along the end-to-end transmission path such that a packet is modeled as being passed between groups of nodes. Each group of nodes can be considered a *stage* in the transmission path. Figure 5 illustrates this on a *wide path* topology between a source and destination where the nodes lie on a regular grid and each stage, except the first and last, consists of 3 nodes. We define the *path width at a stage* as the number of nodes in the stage.

DSF does not use acknowledgments to signal a node to retransmit a packet on a failed link. This is because existing MAC layers such as that in IEEE 802.15.4 [4] usually do not support acknowledgments on broadcasts and multicasts. In fact, IEEE 802.15.4 does not even have built-in mechanisms to support multicast. Instead, multicast would need to be implemented indirectly by adding another layer above the MAC to filter out broadcasts that are not from a predefined set of nodes. Clearly, it would be complicated to acknowledge a packet in this scheme.

Because DSF does not retransmit packets, with careful scheduling consecutive packets will not queue in the network if there is only a single source transmitting to a single sink. In the case of multiple flows (source and sink pairs), queuing may

still be necessary. In networks with multiple flows, we may still be able to apply the single-source-single-sink model developed in the following subsection to regions of the network where separate flows do not overlap. Again, the implications of this assumption is discussed in Section IV-D.

Our DSFMC model of DSF requires the *sets* of link transmissions between *distinct* pairs of stages to be independent. Like UPD, DSF uses frequency hopping over time to help justify this assumption. However, the model allows the link transmissions between the *same* pair of stages to be correlated. This mirrors reality because on any single multicast transmission, all the receiving nodes are listening on the same channel.

Also, in our DSFMC model we assume that all nodes in one stage transmit their copy of the packet before the nodes in the next stage transmit their copy of the packet. It is conceivable that you can minimize the end-to-end latency of a single packet by transmitting the packet in the next stage immediately after the first successful reception. However, if a source is generating a stream of packets, you cannot reduce the *average* latency of the packets without adding a mechanism to eliminate redundant transmissions. The analogy is that of a water pipe — the rate at which you can take out *all* the water in the pipe is limited by the rates at the ends of the pipe, regardless of the design of the pipe in the middle. Designing a clever mechanism to eliminate redundant transmissions so as to decrease latency while keeping the same reliability is beyond the scope of this paper.

Our model also assumes that the transmissions of nodes within a stage will interfere with each other, so they must be scheduled in separate time slots. We make this assumption because most sensor network nodes have only one radio and can only listen to one channel at a time. It is conceivable that because of long and short links, the later stages can consist of nodes so widely separated in space such that the transmissions do not interfere with each other and can be scheduled simultaneously. These types of routing schedules are typically not ideal since this means that only a few nodes in the next stage can hear any one transmission in the previous stage, meaning they are not taking full advantage of the multicast nature of the wireless medium. Therefore, they will not be considered in our model.

In DSF routing schedules, a node can be shared between multiple stages. One way to define stage membership is to put a node in stage k if it has a path of length k to the source node (See Figure 6). Like UPD, we assume that the links in the routing topology for DSF do not form a cycle.

Complications arise when sharing nodes between stages because unlike flooding, staged flooding puts the constraints that a packet can only be transmitted from a node if it received the packet *prior* to the time another node in its stage first transmits. Consider the routing topology in Figure 6 and assume that at time 1, node 2 has a copy of the packet and node 3 does not. At time 2, assume node 2 broadcasts the packet and node 3 receives it. Unlike the typical notion of flooding, in staged flooding, node 3 *cannot* transmit the packet at time 3 because it did not receive it at time 1, the time slot at the beginning of stage 1. This idiosyncrasy is necessary for the DSFMC model developed below to hold. To

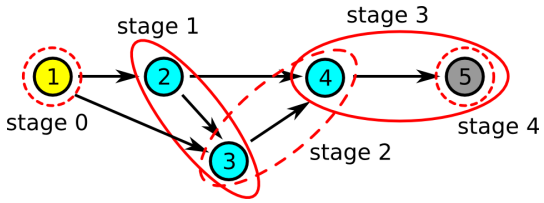

Fig. 6. Directed staged flooding example corresponding to Equation 16.

enforce this condition, packets may carry with them a field indicating during which stage they were last transmitted. Of course, forcing a node to not transmit a packet on the next scheduled time slot for staged flooding may result in a worse $p_{net}^{(t_d)}$ than flooding.

Sharing a node i between multiple stages k and $k+1$ also raises the issue of whether node i should erase a packet after a transmission/multicast in stage k or retain the packet for stage $k+1$. Erasing the packet after one transmission would allow for a simpler implementation and matches the behavior of nodes not shared between stages. However, if we knew that node i is part of stage $k+1$ and is scheduled to transmit the packet again, we can get a better $p_{net}^{(t_d)}$ by retaining the packet. This is akin to a "self-transmission" from node i to itself with probability 1. We will assume the latter in our examples in this paper.

The goals of our DSFMC model is the same as that of the MTMC model: find the end-to-end connectivity of the network $p_{net}^{(t_d)}$, identify hot spots, and find the robustness of the calculations to link probability modeling uncertainty.

B. Directed Staged Flooding Markov Chain Model

As before, we represent the routing topology as a graph $G = (\mathcal{V}, \mathcal{E})$ and denote a node in the network as $i \in \mathcal{V} = 1, \ldots, N$ and a link in the network as $l \in \mathcal{E} \subset \{(i,j) \,|\, i,j \in \mathcal{V}\}$, where $l = (i,j)$ represents a link transmitting from node i to node j. Because each link is used only once when transmitting a single packet, the link success probability for link $l = (i,j)$ is treated as being time-invariant and is denoted p_l, or p_{ij}.

As mentioned earlier, one method of partitioning the nodes into stages is to put a node in stage k if it has a path of length k to the source node a. Note that given the adjacency matrix A of a routing topology G, the number of walks from a node i to a node j in G with length k is $(A^k)_{ij}$, where a walk is a path that is permitted to use vertices more than once [23]. However, the walks in G are paths because G is a directed acyclic graph. Therefore, if $(A^k)_{aj} \neq 0$, then node j belongs to stage k. Of course, there are other methods to partition nodes into stages. The choice of how to partition the nodes strongly affects the choice of a transmission schedule.

The main difference of the DSFMC model from the MTMC model lie in the definition of the states. Here, a state in the Markov chain at a stage represents the *set* of nodes in the stage that successfully received a copy of the packet. The transition probabilities between the states depend on the joint probability of successful link transmissions between stages.

Directed Staged Flooding Markov Chain Model Let's assume we have a routing topology with $K+1$ stages $0, \ldots, K$.

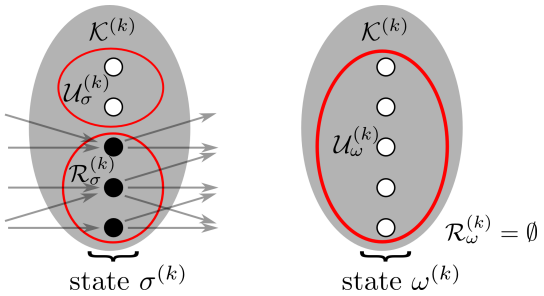

Fig. 7. Mapping of states to nodes that received a packet in the DSFMC model. On the left is an example of a state $\sigma^{(k)}$ and on the right is the state $\omega^{(k)}$ where no packets have been received.

Each stage k has N_k nodes, and the set of 2^{N_k} possible states in stage k is represented by the set of numbers $\mathcal{S}^{(k)} = \{0, \ldots, 2^{N_k} - 1\}$. Let $\mathcal{K}^{(k)}$ be the set of nodes in stage k and for each state $\sigma^{(k)} \in \mathcal{S}^{(k)}$, let $\mathcal{R}_\sigma^{(k)} \subset \mathcal{K}^{(k)}$ be the set of nodes that have received a copy of the packet and $\mathcal{U}_\sigma^{(k)} = \mathcal{K}^{(k)} \backslash \mathcal{R}_\sigma^{(k)}$ be the set of nodes that have not received a copy of the packet (See Figure 7). Let $\omega^{(k)}$ denote the state where no nodes received a copy of the packet in stage k.

Let $R_\sigma^{(k)}$ denote the event that only the nodes in $\mathcal{R}_\sigma^{(k)}$ received a copy of the packet, $S_{(i,j)}$ denote the event a packet was at node i and link (i,j) successfully transmitted the packet, and $\bar{S}_{(i,j)}$ denote the event that a packet was at node i but link (i,j) failed.[4] The conditional probability of the next state $\mathbf{X}^{(k+1)}$ being in state $\sigma^{(k+1)}$ given that the current state $\mathbf{X}^{(k)}$ is $\sigma^{(k)}$ can be expressed in terms of these events as

$$\mathbb{P}(\mathbf{X}^{(k+1)} = \sigma^{(k+1)} | \mathbf{X}^{(k)} = \omega^{(k)}) = \begin{cases} 1 & : \quad \sigma^{(k+1)} = \omega^{(k+1)} \\ 0 & : \quad \text{otherwise} \end{cases}$$

if $\sigma^{(k)} \neq \omega^{(k)}$

$$\mathbb{P}(\mathbf{X}^{(k+1)} = \sigma^{(k+1)} | \mathbf{X}^{(k)} = \sigma^{(k)}) = \mathbb{P}\left(\bigcap_{u^{(k+1)} \in \mathcal{U}_\sigma^{(k+1)}} \left(\bigcap_{r^{(k)} \in \mathcal{R}_\sigma^{(k)}} \bar{S}_{(r^{(k)}, u^{(k+1)})} \right) \cap \bigcap_{r^{(k+1)} \in \mathcal{R}_\sigma^{(k+1)}} \overline{\left(\bigcap_{r^{(k)} \in \mathcal{R}_\sigma^{(k)}} \bar{S}_{(r^{(k)}, r^{(k+1)})} \right)} \,\middle|\, R_\sigma^{(k)} \right) \tag{12}$$

where the overbar denotes taking the complement of an event. The transition probability matrices between stage k and $k+1$ are $P^{(k+1)} \in [0,1]^{N_{k+1} \times N_k}$, where the entry in position $(\sigma^{(k+1)}, \sigma^{(k)})$ of the matrix is $\mathbb{P}(\mathbf{X}^{(k+1)} = \sigma^{(k+1)} | \mathbf{X}^{(k)} = \sigma^{(k)})$.

The initial state $\mathbf{X}^{(0)}$ is the state $\sigma^{(0)}$ corresponding to $\mathcal{R}_\sigma^{(0)} = \{a\}$, where a is the node sending the initial packet. Then, the probability distribution $\mathbf{p}^{(k)} \in [0,1]^{N_k}$ of the state at stage k is

$$\mathbf{p}^{(k)} = \underbrace{P^{(k)} \cdots P^{(2)} P^{(1)}}_{P^{(k)}} \mathbf{p}^{(0)} \tag{13}$$

■

[4]The event $S_{(i,j)}$ is empty (and occurs with probability 0) if link (i,j) does not exist.

Equation 12 describes state transitions between stages in terms of the success and failure of links incident on each receiving node. The event that a node $u^{(k+1)} \in \mathcal{U}_\sigma^{(k+1)}$ does not receive a copy of the packet is the intersection of the events where all the incoming links from the nodes in the previous stage with a copy of the packet fail. The event that a node $r^{(k+1)} \in \mathcal{R}_\sigma^{(k+1)}$ receives a copy of the packet is the *complement* of the intersection of the events that all the incoming links from the nodes in the previous stage with a copy of the packet fail. The event that stage k is in state $\sigma^{(k)}$ and stage $k+1$ is in state $\sigma^{(k+1)}$ is the intersection of all these events.

In the special case where the links are independent, the probability of the joint events can be factored into a product of the probabilities of individual link transmissions:

$$\mathbb{P}(\mathbf{X}^{(k+1)} = \sigma^{(k+1)} | \mathbf{X}^{(k)} = \omega^{(k)}) = \begin{cases} 1 & : \quad \sigma^{(k+1)} = \omega^{(k+1)} \\ 0 & : \quad \text{otherwise} \end{cases}$$

if $\sigma^{(k)} \neq \omega^{(k)}$

$$\mathbb{P}(\mathbf{X}^{(k+1)} = \sigma^{(k+1)} | \mathbf{X}^{(k)} = \sigma^{(k)}) = \left(\prod_{\substack{u \in \mathcal{U}_\sigma^{(k+1)} \\ i \in \mathcal{R}_\sigma^{(k)}}} (1 - p_{iu}) \right) \prod_{r \in \mathcal{R}_\sigma^{(k+1)}} \left(1 - \prod_{i \in \mathcal{R}_\sigma^{(k)}} (1 - p_{ir}) \right) \tag{14}$$

Note that the model is described in terms of stages, not time. Assuming that the nodes of a stage transmit sequentially on separate time slots, a stage k can be converted to a time t measured in units of time slots by the equation $t = \sum_{i=0}^{k-1} N_i$. Therefore, assuming that only the nodes in stage $K-1$ transmit to the destination, if we let b be the state in stage K where the destination receives a copy of the packet, we have

$$p_{net}^{(t_d)} = \begin{cases} 0 & : \quad t_d \leq \sum_{i=0}^{K-2} N_i \\ \mathbf{p}_b^{(K)} & : \quad t_d \geq \sum_{i=0}^{K-1} N_i \end{cases} \tag{15}$$

and $0 \leq p_{net}^{(t_d)} \leq \mathbf{p}_b^{(K)}$ when $\sum_{i=0}^{K-2} N_i < t_d < \sum_{i=0}^{K-1} N_i$.

If stages besides $K-1$ transmit to the destination, we would need to modify the DSFMC model to calculate $p_{net}^{(t_d)}$. We would need to add the destination node to all the stages and add a "self-transmission" link of probability 1 to the destination node before calculating the transition matrices $P^{(k)}$ between each pair of consecutive stages.

Finally, note that except in the special case where there exists a path through the network from the source to the destination with end-to-end connectivity 1, $p_{net}^{(t_d)} < 1$ for all t_d. All copies of a packet can be lost in the network because we do not use acknowledgments and retransmissions to guarantee a copy of the packet has been delivered.

C. DSFMC Examples and Discussion

As an example, let's consider the stages with path width 3 in Figure 5. Assume the links are independent, that each link has the same transmission success probability p, and let $\bar{p} = 1 - p$. Then, the probability that a node in stage $k+1$

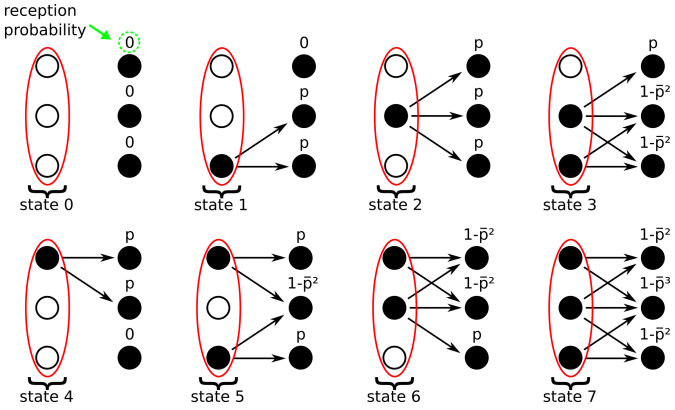

Fig. 8. Markov chain states for the routing topology in Figure 5, excluding the states for the source and the destination.

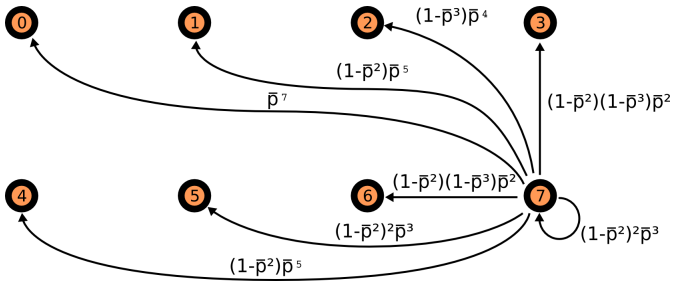

Fig. 9. Markov chain transition diagram for a stage of path width 3 in the routing topology in Figure 5. Here, only the outgoing transitions and associated transition probabilities from state 7 are shown.

receives a copy of the packet given the state of stage k is 1 minus the product of incoming link failure probabilities, as shown in Figure 8. The transition probability between states can be obtained by applying Equation 14. Figure 9 illustrates the transitions out of state 7 (The full 8×8 transition matrix can be found in Appendix C).

Note that if the number of nodes in each stage vary, the dimensions of the state probability distribution vector $\mathbf{p}^{(k)} \in [0,1]^{2^{N_k}}$ vary with time as the copies of the packet are transmitted between stages. This is the case for stages involving the the source and destination nodes of Figure 5.

In the example of Figure 6, not only do the dimensions of the state probability distribution vector vary with time but also some of the nodes are shared between stages. To represent the state at each stage k, we first order the nodes in each stage from smallest to largest node id and re-index them from $0, \ldots, N_k - 1$. Then, for each node with a new index n we set $i_n = 1$ if the node has a copy of the packet and $i_n = 0$ otherwise. The state is then just $\sigma^{(k)} = \sum_{n=0}^{N_k-1} i_n 2^n$. Assuming the links are independent, the equations that describe the DSFMC model are

$$P^{(1)} = \begin{bmatrix} 1 & \bar{p}_{12}\bar{p}_{13} \\ 0 & p_{12}\bar{p}_{13} \\ 0 & \bar{p}_{12}p_{13} \\ 0 & p_{12}p_{13} \end{bmatrix} \quad P^{(2)} = \begin{bmatrix} 1 & \bar{p}_{23}\bar{p}_{24} & 0 & 0 \\ 0 & p_{23}\bar{p}_{24} & \bar{p}_{34} & \bar{p}_{24}\bar{p}_{34} \\ 0 & \bar{p}_{23}p_{24} & 0 & 0 \\ 0 & p_{23}p_{24} & p_{34} & (1-\bar{p}_{24}\bar{p}_{34}) \end{bmatrix}$$

$$P^{(3)} = \begin{bmatrix} 1 & \bar{p}_{34} & 0 & 0 \\ 0 & p_{34} & \bar{p}_{45} & \bar{p}_{45} \\ 0 & 0 & 0 & 0 \\ 0 & 0 & p_{45} & p_{45} \end{bmatrix} \quad P^{(4)} = \begin{bmatrix} 1 & \bar{p}_{45} & 0 & 0 \\ 0 & p_{45} & 1 & 1 \end{bmatrix}$$

$$\mathbf{p}^{(0)} = \begin{bmatrix} 1 & 0 \end{bmatrix}^T \quad P^{(4)} = P^{(4)}P^{(3)}P^{(2)}P^{(1)} \tag{16}$$

where p_{ij} is indexed by the original node ids and again $\bar{p}_{ij} = 1 - p_{ij}$. As mentioned in Section III-A, we assume that if a node i in stage k has a copy of the packet and node i is also in stage $k+1$, then node i will have a copy of the packet in stage $k+1$ with probability 1.

Note that the computational complexity of the DSFMC model is exponential in the path width because the dimensions of the transition probability matrix are exponential in the number of nodes in each stage. This is typically not a problem, because we would want the width of a stage in real deployments to be small (less than 6) to conserve bandwidth and prevent unnecessary flooding of the entire network. If we were to disregard computational complexity and allow the width of each stage to be unbounded, we could model scheduled flooding over the network. Unlike the typical flooding algorithm that may run on a CSMA MAC layer, we would need to impose an order in which the nodes broadcast to construct the model.

D. DSFMC Analysis

1) $p_{net}^{(t_d)}$ for Wide Paths with Repeated Stages: Note that because we developed the DSFMC model for one-to-one routing, it does not make much sense to derive a rate of convergence on $p_{net}^{(t_d)}$ for the entire network. However, for the purposes of choosing a network topology before deployment, it is useful to get a grasp of how $p_{net}^{(t_d)}$ scales as we extend the length K of a wide path topology without having to calculate $p_{net}^{(t_d)}$ for each new network explicitly. We consider the case of a wide path with repeated stages containing a constant number of nodes N_{stage} per stage and the same transition probability matrix $P^{(k)} = P$ between all stages, like the middle stages in the example in Figure 5. For simplicity, the discussion below will ignore the first stage containing the source and the last stage containing the destination.

One approximate way of understanding the gain in end-to-end connectivity and the cost in latency from using Directed Staged Flooding on a wide path is to compare each stage of the path to a node in a single path, and the links between stages to a link on the single path. In the case where the links of the wide path are independent with transmission probability p, we actually transmit N_{stage} times to get the equivalent of $\left|\{(i,j)|i \in \mathcal{K}^{(k)}, j \in \mathcal{K}^{(k+1)}\}\right|$ retransmissions on a single link path with probability p, assuming all the nodes in stage k have a copy of the packet ($\mathcal{U}_\sigma^{(k)} = \emptyset$). In the case of the example in Figure 5, we have a gain of 7 retransmissions for a cost of 3 in latency, assuming $\mathcal{U}_\sigma^{(k)} = \emptyset$.

A better characterization of the tradeoff of end-to-end connectivity with latency t_d (number of stages K) comes from the eigenvalues of P. Assuming we do not have the special case where there exists a path through the network with end-to-end connectivity 1, the DSFMC model has a single recurrent state ω, the state where *no* nodes received a copy of the packet. This means that there is a unique stationary distribution $\mathbf{e}^{[\omega]}$ for the Markov chain model, and $p_{net}^{(t_d)} \xrightarrow{K\to\infty} 0$. The rate of decay of $p_{net}^{(t_d)}$ is given by ρ_*, the magnitude of the largest eigenvalue of P with absolute value strictly less than 1. This is stated more precisely in the following theorem, which is proved in Appendix B.

Theorem 3.1 (DSFMC $p_{net}^{(t_d)}$ converges exponentially to 0): Let $P \in [0,1]^{N\times N}$ be a column stochastic matrix and $\lim_{K\to\infty} P^K \mathbf{p}^{(0)} = \mathbf{e}^{[\omega]}$, where ω is the state where *no* nodes received a copy of the packet. Then

$$p_{net}^{(t_d)} \leq CK^{J-1}(\rho_*)^{K-J+1}, \quad t_d = KN_{stage} \qquad (17)$$

for some constant C dependent on the initial distribution $\mathbf{p}^{(0)}$, $J \in \mathbb{Z}_+$ the size of the largest Jordan block of P, and $\rho_* = \max\{|\lambda| : \lambda \text{ is an eigenvalue of } P \text{ and } |\lambda| < 1\}$. ■

Again, if P is diagonalizable, we get $J = 1$ and Equation 17 becomes

$$p_{net}^{(t_d)} \leq C(\rho_*)^K, \quad t_d = KN_{stage} \qquad . \qquad (18)$$

While this relation is an upper bound, ρ_* is the dominant decay rate for large K because all the eigenvectors of P with eigenvalue magnitudes less than 1 decay exponentially with K. Thus, one can use ρ_* to compare wide paths with repeated stages of different widths and quickly assess the tradeoff between reliability and latency. Unfortunately, this type of analysis cannot apply to paths where $P^{(k)}$ varies with stage k.

2) Traffic Distribution: We can obtain the probability that a copy of the packet is at a node i at time t directly from our model by translating t to k and looking at $\sum_{\{\sigma^{(k)}|i\in\mathcal{R}_\sigma^{(k)}\}} \mathbb{P}(\mathbf{p}^{(k)} = \sigma^{(k)})$. To get a sense of how robust the network is to the failure/compromise of a node i, we can just remove i from the routing graph and recalculate the end-to-end connectivity $p_{net}^{(t_d)}$. Alternatively, in the same sense as Section II-D2, we can also calculate the probability that a copy of the packet visits a node i at or before time t, $\alpha_i^{(t)}$. To do this we remove all the outgoing edges of i, add a "self transmission" link of probability 1 from node i to itself over all time slots, and compute $\alpha_i^{(t)} = \sum_{\{\sigma^{(k)}|i\in\mathcal{R}_\sigma^{(k)}\}} \mathbb{P}(\tilde{\mathbf{p}}^{(k)} = \sigma^{(k)})$, where $\tilde{\mathbf{p}}^{(\mathbf{k})}$ is the state probability distribution on the modified routing schedule and topology.

3) Link Perturbation in Topology with Independent Links: Let us consider a DSFMC model where the links are independent. As before, we would like to know the sensitivity of $p_{net}^{(t_d)}$ to errors in estimating a link probability p_l. Recall that unlike UPD, there are no link retransmissions in DSF. As we will see below, on routing schedules where nodes are not shared between stages this implies that $p_{net}^{(t_d)}$ is a linear function of the single-link estimation error δ. Thus, we can compute bounds on the actual end-to-end connectivity $\hat{p}_{net}^{(t_d)}$ using the maximum and minimum possible values of the real link probability $\hat{p}_l$, $p_l + \epsilon$ and $p_l - \epsilon$ respectively.

To show that $p_{net}^{(t_d)}$ is a linear function of δ, note that in Equation 14, the transition probability between states in adjacent stages are a linear function of the individual link probabilities (the probability associated with a link appears in the expression once). This means that the transition matrices $\hat{P}^{(k)}$ are a linear function of each link probability p_l. Also, each link probability p_l appears in only one matrix $\hat{P}^{(k)}$ because each link is used only once to transmit a packet. This is because there are no retransmissions in the network and no nodes are shared between stages, so no node will transmit more than once when routing a single packet through the network. As a result, $\hat{P}^{(\underline{K})}$ is also a linear function of p_l. Finally, $p_{net}^{(t_d)}$ is a linear function of $\hat{P}^{(\underline{K})}$ and hence also a linear function of p_l, meaning it is a linear function of δ.

In fact, because wide path routing allows for multiple copies of the packet in the network, packets do not get "trapped" at a node like the example in Figure 4. This means $p_{net}^{(t_d)}$ increases with increasing p_l for any link l. Another way to see this is to realize that $p_{net}^{(t_d)}$ for DSF is actually the sum of the probabilities of a disjoint set of events, where each event represents successful delivery of the packet along a distinct path between the source and destination. Each of these "path events" is the intersection of successful link transmission events (and not any link failure events, as would be the case if we had retransmissions). As a result, increasing a link probability can only increase the probabilities of the path events, which increases $p_{net}^{(t_d)}$.

IV. UPD and DSF Comparisons

Qualitatively, the main difference between UPD and DSF is the technique they use to provide reliable end-to-end packet delivery. Both employ frequency diversity to get independent links and get spatial diversity by establishing multiple paths to the sink. However, UPD only retransmits the packet on link failures while DSF uses multicasting and a fixed number of "preemptive retransmissions" at each stage for reliability. We would like to get a sense of the conditions under which one type of routing is better than the other, and for qualitative comparisons we use the example of routing on a wide path grid, where the width of the path is the number of rows and the length of the path is the number of columns. The metric used in the comparisons is the end-to-end connectivity as a function of latency, $p_{net}^{(t_d)}$, computed using the MTMC and DSFMC models presented in the previous sections.

For a fair comparison, we choose a routing topology where every node in one column of a grid (a stage in DSF) can route to every other node in the next column with equal, independent link probabilities p_l. To accommodate interference assumptions of an isotropic/disk radio model and be able to schedule the transmission of *unique* packets closely in time, we would space the columns of the grid much further apart than the rows of the grid. This topology makes the choice of an optimal UPD routing schedule easier and minimizes the "edge effects" of routing topologies like Figure 5, where the nodes at the top and bottom of the grid transmit to fewer other

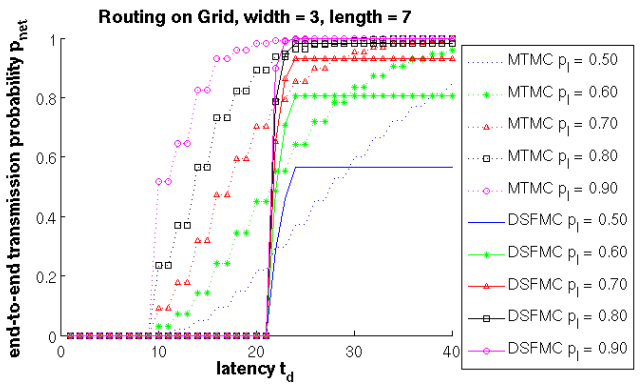

Fig. 11. End-to-end connectivity as a function of latency for varying link probabilities using the routing schedules described in Figure 10.

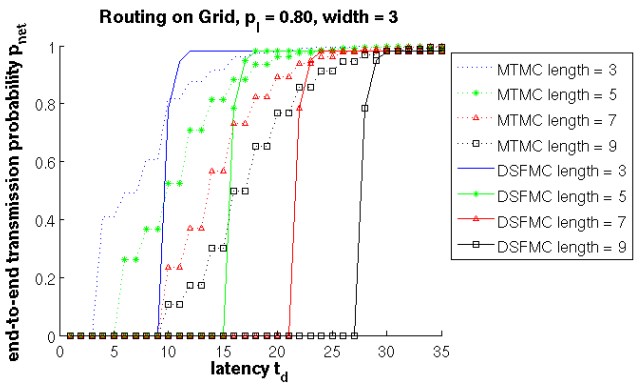

Fig. 12. End-to-end connectivity as a function of latency for varying path lengths using the routing schedules described in Figure 10.

nodes in the next column than the nodes in the middle of the grid. The routing schedule for Directed Staged Flooding and Unicast Path Diversity is described in Figure 10 for paths of width 3.

Also, for all our plots, we assume that the time to send an acknowledgment for UPD is negligible and can be sent back in the same time slot as the original transmission. For an 802.15.4 radio, the time for a minimal ACK packet is $\frac{(6+25Bytes/pkt)(8bits/Byte)}{250kbps} \approx 0.99ms$, where the packet has a 6 Byte PHY header and a 25 Byte MAC header. This is small relative to a large data packet, which can be as large as 131 Bytes. Of course, if one wishes to reinterpret the results in the figures in this section assuming that acknowledgments cause the time slots for UPD to be larger than the time slots for DSF, he would just scale the time on the UPD plots accordingly.

A. End-to-end Connectivity Comparisons

Figure 11 compares $p_{net}^{(t_d)}$ of the two routing schemes under a range of different link probabilities.[5] UPD has the potential to deliver packets from the source to the sink in a shorter period of time, but the packet delivery time has a larger variance. Note that for lower link probabilities, there is clearly a range of arrival times where DSF provides better end-to-end connectivity than UPD. However, because $\lim_{t\to\infty} p_{net}^{(t_d)} = 1$ for UPD and p_{net} for DSF is a fixed value strictly less than 1 after the last stage transmits (assuming $p_l \neq 1$), UPD can always provide better end-to-end connectivity at high latencies t_d.

Naturally, in Figure 12 we see that a larger path length tends to favor UPD over DSF. The range of arrival times when DSF provides better end-to-end connectivity than UPD becomes shorter, and the difference in p_{net} of the two schemes at the time when the last stage in DSF finishes transmission is smaller. The times at which p_{net} for UPD exceeds that of DSF after the last stage transmits is given in Table I. Recall from the discussion from Section III-D1 that as the number of stages increases, the end-to-end connectivity under Directed Staged Flooding approaches 0. However, even for a path width of 3, the rate at which p_{net} approaches 0 is small, as seen by the last column of Table I. To a rough approximation, an increase in path length seems to linearly increase the delay in packet delivery for the range of parameters considered in Table I.

path length	minimum t_d where UPD $p_{net}^{(t_d)} >$ DSF $p_{net}^{(t_d)}$	UPD $p_{net}^{(t_d)}$	DSF $p_{net}^{(t_d)}$
3	20	0.98292	0.98173
5	24	0.98666	0.98226
7	28	0.98747	0.98226
9	32	0.98886	0.98225

TABLE I
$p_{net}^{(t_d)}$ CROSS OVER POINT FOR DSFMC AND MTMC GRAPHS IN FIGURE 12.

Figure 13 shows how $p_{net}^{(t_d)}$ for DSF increases with path width, and plots $p_{net}^{(t_d)}$ for UPD of different widths for comparison. At first glance, the graphs are striking because it shows that UPD on paths of width 3 always perform better than UPD on paths of width 5. This is because our MTMC model assumes that retransmissions between a pair of nodes are independent whereas in reality they may be slightly correlated, even if the retransmission is on a different frequency. As a result, in the MTMC model retransmission to a node is just as good as transmitting to another neighboring node when computing $p_{net}^{(t_d)}$. What *is* captured in the MTMC model is the extra time necessary to schedule transmissions to the sink from a wider path, which results in a longer time for the packet to reach the destination. Therefore, wider paths in the MTMC model actually perform worse in our calculations. The same argument also explains why in Figure 13 UPD on paths of width 3 always performs better than DSF on paths of width 5. The benefits of using a wider path in UPD will be evident when we consider the robustness of $p_{net}^{(t_d)}$ to node compromise or link probability estimation error.

When designing networks, we can increase the end-to-end connectivity p_{net} by using UPD and waiting longer periods for packets, or we can increase the number of paths from source to destination when using DSF. Of course, we can increase the number of paths *and* use UPD, but as Table II and Figure 13 show, there are ranges of latencies where DSF outperforms UPD on the same routing topology. In the grid/wide path

[5]Note that in this and subsequent plots, we perform the DSFMC calculations at the time granularity of time slots, not stages, unlike the description of Equation 15 in Section III-B.

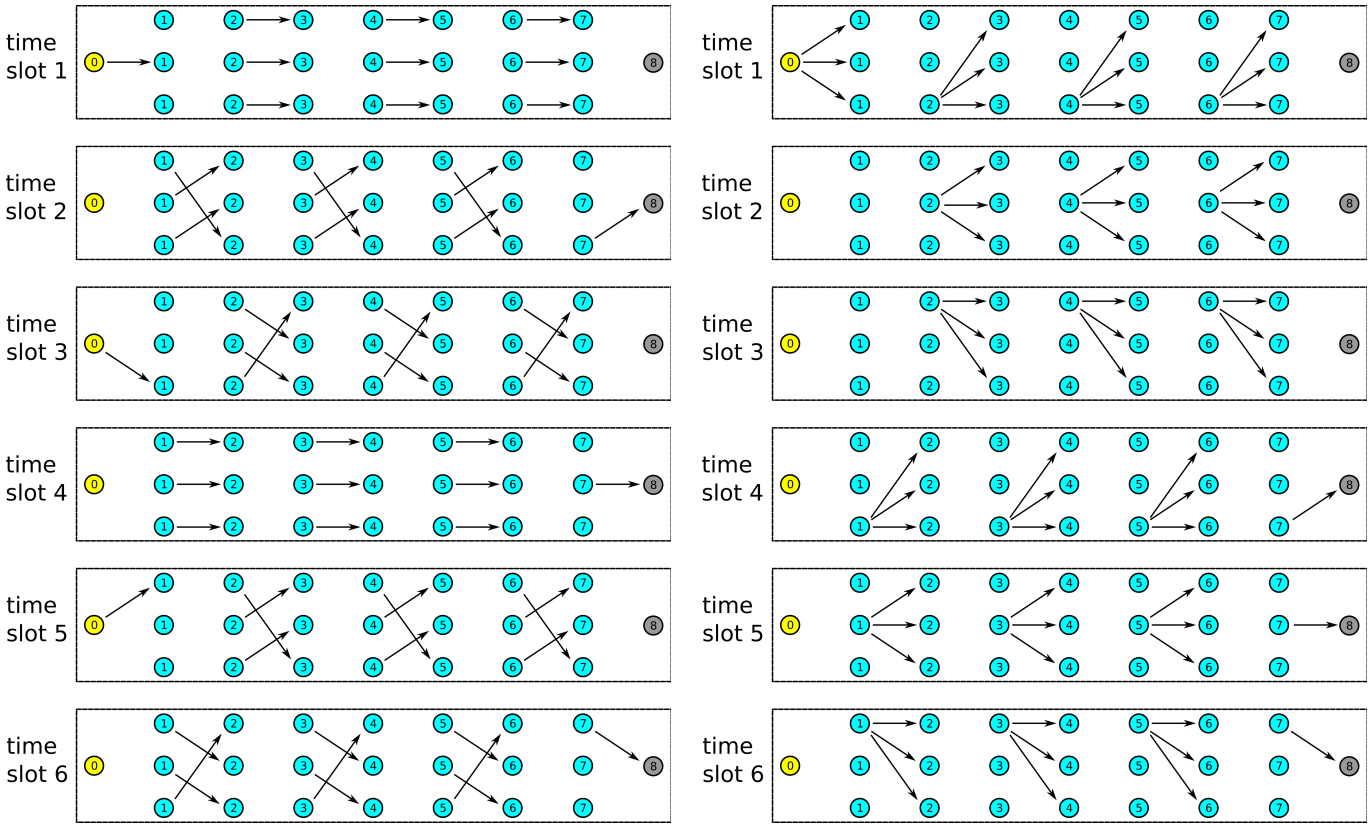

Fig. 10. (left) UPD and (right) DSF schedules for routing on a grid of width 3, used in the calculations for the graphs in Section IV.

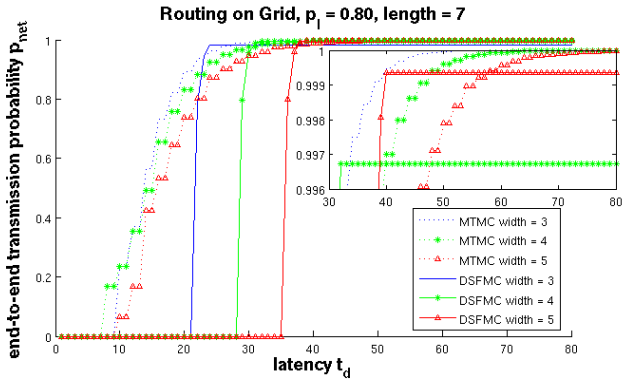

Fig. 13. End-to-end connectivity as a function of latency for varying path widths using the routing schedules described in Figure 10, with magnification of plot for p_{net} near 1.

path width	minimum t_d where UPD $p_{net}^{(t_d)}$ > DSF $p_{net}^{(t_d)}$	UPD $p_{net}^{(t_d)}$	DSF $p_{net}^{(t_d)}$
3	28	0.98747	0.98226
4	40	0.99699	0.99672
5	58	0.99942	0.99936

TABLE II
$p_{net}^{(t_d)}$ CROSS OVER POINT FOR DSFMC AND MTMC GRAPHS IN FIGURE 13.

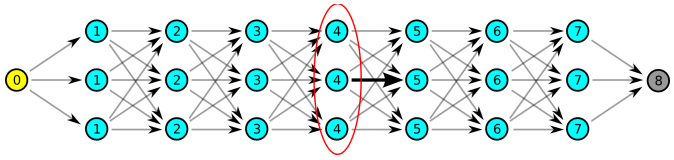

Fig. 14. The width 3 routing topology used for studying traffic distribution and sensitivity to link estimation error. Figure 15 studies the traffic distribution of nodes in the middle stage $\mathcal{K}$ (stage 4) of the path, circled in red. Figures 16 and 17 study sensitivity to link estimation error on the link in the center of the middle stage, highlighted by bold/darker print.

routing topology with a path width of 5 UPD needs 45% more latency than DSF to get better end-to-end connectivity. Note that the difference in performance of DSF and UPD from increasing path width are lessened if we were to consider a routing topology where a node can only communicate to a subset of the nodes in the next stage.

B. Robustness Comparisons

An even distribution of packet traffic over the nodes in the network lessens the formation of "hot spots" in the network, nodes whose undetected failure or compromise greatly impact the end-to-end connectivity of the network. Using the techniques described in Sections II-D2 and III-D2, we compute the traffic distribution α on a group of nodes $\mathcal{K}$ in our grid routing topology that cut the routing graph between the source and destination, as depicted in Figure 14. In UPD, because there is one copy of the packet in the network and none of the nodes in $\mathcal{K}$ route the packet to each other, $\sum_{i\in\mathcal{K}} \alpha_i = 1$. This is not true in DSF because there are multiple copies of the packet in the network.

We expect wider paths to distribute the traffic more evenly

path width	DSF Δp_{net}
3	1.5942e-05
4	1.1627e-07
5	1.0757e-09

TABLE III
Δp_{net} AFTER THE LAST STAGE TRANSMITS IN DSF, CORRESPONDING TO THE RIGHT GRAPH IN FIGURE 16.

among the nodes in $\mathcal{K}$. However, traffic distribution in UPD is highly dependent on the schedule and link probabilities. For instance if the link probabilities in the network are lower, you expect that UPD would have to try more links to reach the destination and thus spread the traffic through the network more evenly. Even with the simple, regular schedule show in Figure 10 on topologies of width 5 with a fairly low link probability $p_l = 0.8$, we see in Figure 15 that UPD does not distribute the packets completely evenly over the nodes in $\mathcal{K}$. On the other hand, DSF tends to spread copies of the packet over the nodes in $\mathcal{K}$ better than UPD for all path widths because it multicasts the packets.

In the same sense, we expect DSF to be more robust than UPD to link estimation error because it multicasts packets and thus tends to spread packets over more paths. Despite the argument in Section II-D3 that in general we cannot simply substitute perturbed link probabilities to calculate the effect of link perturbation on end-to-end connectivity in UPD, we find that for the simple, regular routing schedule exemplified in the left diagram of Figure 10, substituting single link perturbations provides rather predictable effects on the end-to-end connectivity. This is shown on the left graph in Figure 16 for perturbations on the link identified in Figure 14 (This is the single link that we perturb for the studies in this section, which is typical of other links in the middle of the path because of the regular structure of our schedule and topology). Comparing this with the right graph of Figure 16 confirms that DSF is orders of magnitude less sensitive to link perturbation than UPD. Of course, because $\lim_{t\to\infty} p_{net}^{(t_d)} = 1$, the end-to-end connectivity of UPD on routing schedules with one sink will eventually be less sensitive than that of DSF. In our example, it takes 46 time slots before the change in UPD end-to-end connectivity from a link perturbation of $\epsilon = 0.1$ is less than that of DSF ($\Delta p_{net}^{(46)} = 2.4778 \times 10^{-6}$ for UPD and $\Delta p_{net}^{(46)} = 4.9071 \times 10^{-6}$ for DSF).

If we look at grid topologies with larger width, we see from Figure 17 that $p_{net}^{(t_d)}$ for both UPD and DSF are much less sensitive to link perturbations. In fact, we see from Table III that in our routing examples the sensitivity to single link estimation errors from DSF drops by four orders of magnitude when we move from topologies of width 3 to width 5.

C. Other Considerations

To make fair comparisons between the performance of UPD and DSF, one needs to select optimal, or close to optimal schedules for both routing algorithms. Scheduling for UPD may be particularly tricky, and choosing an inefficient UPD schedule can result in a significantly worse $p_{net}^{(t_d)}$. Some examples illustrating this can be found in Appendix D.

Another point of comparison mostly ignored in our discussion is the power consumption of UPD and DSF, particularly in sensor networks where nodes can be scheduled to sleep (go into low power mode). In UPD, if a pair of nodes is scheduled to communicate but the receiver does not hear the preamble of a packet at the beginning of a time slot, the receiver can assume that the transmitter does not have a packet and go to sleep for the remainder of the time slot, saving power. Receivers in DSF can do the same but because there are multiple copies of a packet in the network, there are less opportunities to sleep. This can be a problem particularly for current generation IEEE 802.15.4 radios such as the CC2420, where the current drawn by the receiver is actually *higher* than the current drawn by the transmitter [24].

Finally, the traffic distribution calculations are also useful for selecting good routing schedules for UPD. How to calculate the traffic distribution to compare schedules is subtle because the traffic distribution for UPD also depends on when (which time slot in the superframe) the packet is ready for transmission from the source. In reality, the period at which packets are generated from a source may not match the length of the superframe, and packets may be ready for transmission mid-frame. Take the example of the UPD routing schedule in Figure 10, and for the sake of argument assume $p_l = 1$ (or very close to 1). If the packet is transmitted from the source on time slot 1, then it will traverse through the nodes on the upper half of the grid. But if the packet is transmitted on time slot 3, then it will traverse through the lower half of the grid. We can take advantage of this to spread consecutive packets along different paths so that in reality, where there is queuing in the network, packets are less likely to queue at a node midway between the source and the destination. Depending on the application, it may be more important to select schedules that are less likely to queue packets in the network, even if it means that for any individual packet the probability distribution of paths it takes to reach the destination is spread less evenly over the nodes in the network.

D. Communication Tradeoffs for Control Systems

If we wish to use UPD or DSF for control systems, we need to establish a routing topology from the controller to the sensors and actuators, and a routing topology from the sensors and actuators back to the controller. For UPD, this means establishing two routing graphs, one many-to-one graph rooted at the controller for collecting observations from the sensors and one one-to-one graph rooted near the actuators (assuming they are together) and receiving commands from the controller. Because there are now two routing topologies on the same network, the transmissions for the two topologies must be scheduled jointly, which may result in higher latencies. Joint scheduling is necessary to prevent two links from being scheduled on the same channel and time slot. This is of concern if the paths in the two topologies are not disjoint and we assume that each node has one radio and can only listen to one neighbor at a time. This problem gets worse if we wish to have multiple control loops over the same wireless network.

As mentioned in Sections III-A and II-A, we do not model queuing in the network in our MTMC and DSFMC models. In

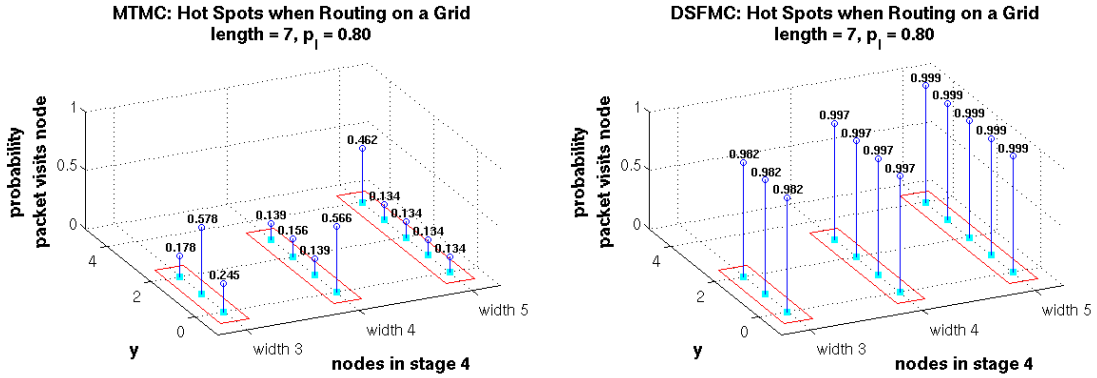

Fig. 15. Traffic distribution of nodes in the middle stage $\mathcal{K}$ (See Figure 14) of routing topologies of varying widths. Note that due to errors in rounding, the probabilities for the middle stages in MTMC may not add exactly to 1. These graphs use the routing schedules described in Figure 10.

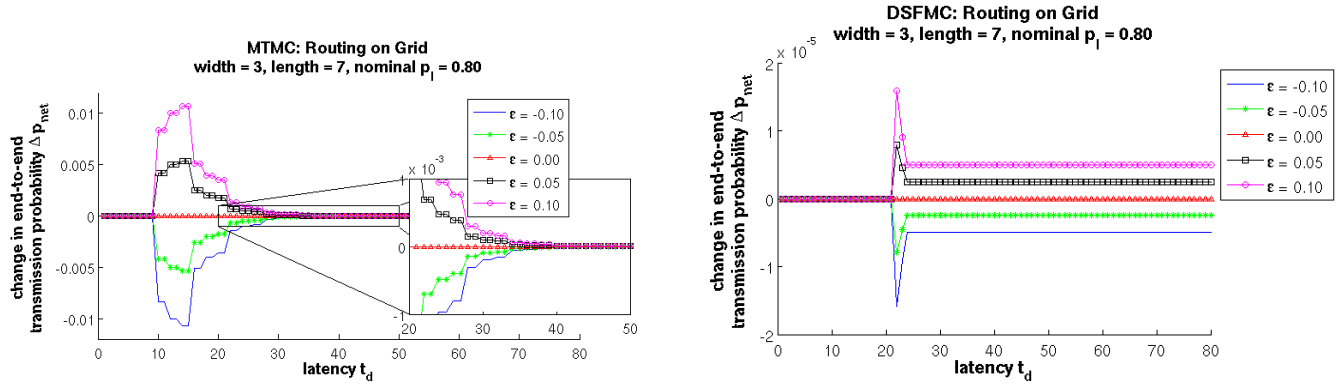

Fig. 16. Change in end-to-end connectivity as a function of latency for link perturbations of varying magnitude ϵ. The MTMC graph is magnified for easier comparison with the DSFMC graph over the time range of interest. These graphs uses the routing schedules described in Figure 10.

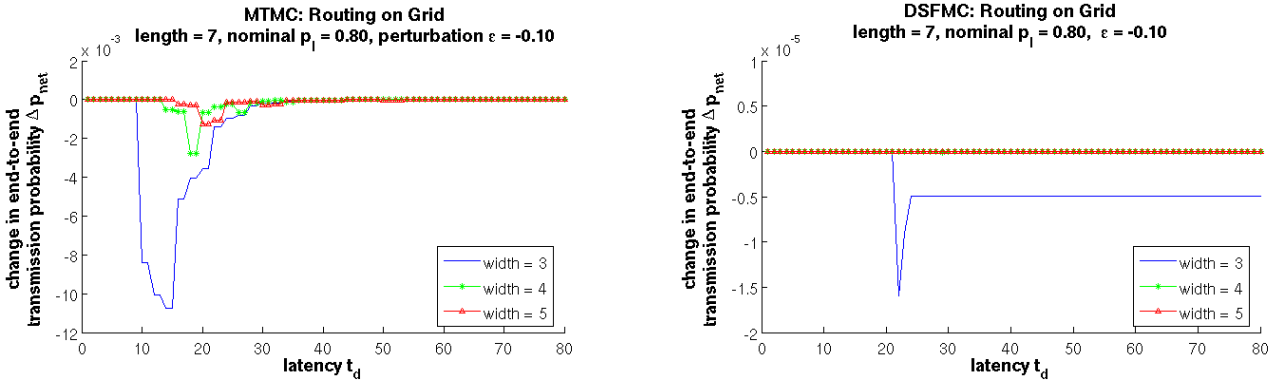

Fig. 17. Change in end-to-end connectivity as a function of latency for link perturbations on routing topologies with different widths. These graphs uses the routing schedules described in Figure 10.

DSF, the transmission schedule is deterministic because there are no retransmissions, and with proper scheduling the packets should not queue in the network. In UPD routing, a packet may queue at a node when a link fails and the packet needs to be retransmitted. This implies that for the MTMC model to hold, we would need to limit the packet rate and choose a routing schedule that is more likely to spread consecutive packets on different paths in the network, taking the lengths and transmission probabilities of the paths into consideration so they do not queue when the paths merge before reaching the destination.

In Section I-A we mentioned that when designing a control application, it may be reasonable to impose a delivery deadline and drop the packet if it takes too long to arrive. In UPD routing, if a packet arrives at a node that has a queued old packet, we can either combine the data in the two packets into one packet or we can drop the older packet and send only the newer packet. The implications of these two schemes is studied in [5].

Using the graphs in Figure 11 and some simple calculations, we can check the feasibility of running a control application on an 802.15.4 wireless network running UPD or DSF using the routing schedules in Figure 10. Assume we have a width 3 path from the controller to the actuators, and a width 3 path from the sensors back to the controller, and all the links have a transmission success probability $p_l = 0.8$. Then after

24 time slots we can get end-to-end transmission probability $p_{net} > 0.95$ between the controller and actuators and between the sensors and controller, both of which are separated by 8 hops. In Dust Network's TSMP 1.0, there are 32 slots a second, which corresponds to ≈ 1.5 seconds round trip time. This round trip time can be decreased in future versions of TSMP because the theoretical limit of an $802.15.4$ radio is $\frac{250kbps}{(6+25+10Bytes/pkt)(8bits/Byte)} \approx 762pkts/sec$ (10 byte payload, 25 byte MAC header and CRC, 6 byte PHY header), resulting in a round trip time of $\approx$ 63ms. Therefore, the types of control applications that we can hope to run on wireless sensor networks spanning 8 hops would have to tolerate round trip latencies on the order of magnitude of tenths of a second under optimal conditions, and seconds if we use current routing algorithms.

V. Conclusions

In this paper, we developed Markov chain models for UPD and DSF routing algorithms that allow us to obtain the end-to-end connectivity of the network as a function of latency, determine the sensitivity of end-to-end connectivity to link probability estimation errors, and determine the robustness of the network to node failure. These models can be very useful for planning a new network deployment given the link probability estimates in the new environment. They can also help guide the selection of routing algorithms and the tuning of their parameters. In the case of UPD, the MTMC model can help in the design of algorithms for selecting routing schedules that optimize for end-to-end connectivity as a function of latency. As seen in Section IV, the choice of routing schedules can vastly affect the performance of the network. In the case of DSF on a wide path, the DSFMC model can help determine the best path width and length for meeting the design constraints.

In order to construct these models for existing networks, the user must have full knowledge of the estimated link probabilities and routing schedule in the network. One possibility is to have a network periodically route back the routing schedule and link probability estimates of all the links in the network, as is done by Dust Network's SmartMesh network manager. Ideally, the time scale over which the link probabilities change is much larger than the time scale for sending a packet through the network with high probability. In some industrial environments, the average channel coherence time was observed to be approximately 0.1 seconds [25], but a more careful study needs to be done on the time scales and the magnitudes over which the link probabilities fluctuate.

Once a user uses the models to characterize the robustness of the network, they can identify regions of the network that may need the addition of more nodes to help distribute traffic and provide redundancy. In wireless networked control systems, if we can calculate $p_{net}^{(t_d)}$ of the network in real-time, we can tune the controller/switch controllers based on the conditions of the network. For instance, in manufacturing we can use an aggressive controller for higher yield when the wireless network is good and a less aggressive controller that does not compromise safety and the quality of the products when the network is bad. We will study in detail the issues of running controllers over mesh wireless networks in an upcoming paper.

Acknowledgment

The authors would like to thank Songhwai Oh, Ian Tan, Kris Pister, and David Tse for offering feedback on the ideas in this paper, and particularly Kris Pister for providing more details about TSMP and 802.15.4 radios.

References

[1] D. Culler, D. Estrin, and M. Srivastava, "Overview of sensor networks," in *IEEE Computer, Special Issue in Sensor Networks*, August 2004.

[2] Industrial Standards and Automation Committee, "ISA-SP100 wireless systems for automation," http://www.isa.org, 2007.

[3] HART Communication Foundation, *WirelessHART Data Sheet*, http://www.hartcomm2.org/hart_protocol/wireless_hart/wirelesshart_datasheet.pdf, 2007, datasheet.

[4] *Part 15.4: Wireless Medium Access Control (MAC) and Physical Layer (PHY) Specifications for Low-Rate Wireless Personal Area Networks (LR-WPANs)*, LAN/MAN Standards Committee of the IEEE Computer Society, 3 Park Avenue, New York, NY 10016-5997, USA, October 2003, 802.15.4 Standard.

[5] L. Schenato, "Optimal estimation in networked control systems subject to random delay and packet loss," in *Proc. of the 45th IEEE Conference on Decision and Control*, December 2006.

[6] B. Sinopoli, L. Schenato, M. Franceschetti, K. Poolla, M. Jordan, and S. Sastry, "Kalman filtering with intermittent observations," *IEEE Transactions on Automatic Control*, September 2004.

[7] J. P. Hespanha, P. Naghshtabrizi, and Y. Xu, "A survey of recent results in networked control systems," *Proceedings of the IEEE*, vol. 95, pp. 138–162, 2007.

[8] W. Weber, J. Rabaey, and E. Aarts, Eds., *Ambient Intelligence*. Springer-Verlag, 2005, ch. TinyOS: An Operating System for Sensor Networks.

[9] J. Hill, R. Szewczyk, A. Woo, S. Hollar, D. Culler, and K. Pister, "System architecture directions for networked sensors," in *ASPLOS-IX*, Cambridge, MA, USA, November 2000.

[10] TinyOS 1.x Documentation, *Multihop Routing*, http://www.tinyos.net/tinyos-1.x/doc/multihop/multihop_routing.html, 2003.

[11] A. Woo, T. Tong, and D. Culler, "Taming the underlying challenges of reliable multihop routing in sensor networks," in *SenSys*, November 5-7 2003.

[12] G. Tolle, "A network management system for wireless sensor networks," Master's thesis, Univ. of California, Berkeley, 2005.

[13] C. Karlof, Y. Li, and J. Polastre, "ARRIVE: Algorithm for robust routing in volatile environments," University of California at Berkeley, Tech. Rep. UCB/CSD-03-1233, May 2002.

[14] S. Dulman, T. Nieberg, J. Wu, and P. Havinga, "Trade-off between traffic overhead and reliability in multipath routing for wireless sensor networks," in *Proceedings of the Wireless Communications and Networking Conference*, 2003.

[15] D. Ganesan, R. Govindan, S. Shenker, and D. Estrin, "Highly-resilient, energy-efficient multipath routing in wireless sensor networks," *SIGMOBILE Mob. Comput. Commun. Rev.*, vol. 5, no. 4, pp. 11–25, 2001.

[16] Dust Networks, Inc., "Technical overview of time synchronized mesh protocol (TSMP)," http://www.dustnetworks.com/docs/TSMP_Whitepaper.pdf, 2006.

[17] C. Intanagonwiwat, R. Govindan, and D. Estrin, "Directed diffusion: A scalable and robust communication paradigm for sensor networks," in *Proc. of 6th Annual International Conference on Mobile Computing and Networks*, August 2000.

[18] A. Nasipuri, R. Castañeda, and S. R. Das, "Performance of multipath routing for on-demand protocols in mobile ad hoc networks," *Mob. Netw. Appl.*, vol. 6, no. 4, pp. 339–349, 2001.

[19] Y. Ganjali and A. Keshavarzian, "Load balancing in ad hoc networks: Single-path routing vs. multi-path routing," in *INFOCOM, 23rd Annual Joint Conference of the IEEE Computer and Communications Societies*, vol. 2, 2004, pp. 1120–1125.

[20] Dust Networks, Inc., *SmartMesh-XT M2030 Product Specification*, http://www.dustnetworks.com/docs/M2030.pdf, 2006, datasheet.

[21] D. Tse and P. Viswanath, *Fundamentals of Wireless Communication*. New York: Cambridge University Press, 2005.

[22] D. P. Bertsekas and J. N. Tsitsiklis, *Introduction to Probability*. Belmont, Massachusetts: Athena Scientific, 2002.
[23] R. A. Horn and C. R. Johnson, *Matrix Analysis*. New York: Cambridge University Press, 1999.
[24] Chipcon Products from Texas Instruments, *2.4 GHz IEEE 802.15.4 / ZigBee-ready RF Transceiver*, http://www.ti.com/lit/gpn/cc2420, March 2007, datasheet, Revision B.
[25] D. Sexton, M. Mahony, M. Lapinski, and J. Werb, "Radio channel quality in industrial wireless sensor networks," in *Proceedings of the ISA/IEEE Sensors for Industry Conference (SIcon)*, February 2005.
[26] H. M. Taylor and S. Karlin, *An Introduction to Stochastic Modeling*, 3rd ed. Academic Press, 1998.
[27] W. J. Stewart, *Introduction to the Numerical Solutions of Markov Chains*. Princeton, New Jersey: Princeton University Press, 1994.
[28] J. S. Rosenthal, "Convergence rates of Markov chains," *SIAM Review*, vol. 37, no. 3, pp. 387–405, 1995.

APPENDIX A
END-TO-END CONNECTIVITY FOR SINGLE PATH WITH RETRANSMISSIONS

Let us consider only a single path between a and b with links $l_1, l_2, \ldots, l_K$. Let $S_k^{(t)}$ denote the event that a packet is at node i and is successfully transmitted on link $l_k = (i,j)$ at time t and $\bar{S}_k^{(t)}$ denote the event that a packet is at node i but link l_k fails at time t. Then, $p_{net}^{(t_d)}$ is just the sum of the probability of a series of disjoint events:

$$\begin{aligned} p_{net}^{(t_d)} = & \\ & \mathbb{P}(S_1^{(1)} \cap S_2^{(2)} \cap \ldots \cap S_K^{(K)}) + \\ & \mathbb{P}(\bar{S}_1^{(1)} \cap S_1^{(2)} \cap S_2^{(3)} \cap \ldots \cap S_K^{(K+1)}) + \\ & \mathbb{P}(S_1^{(1)} \cap \bar{S}_2^{(2)} \cap S_2^{(3)} \cap S_2^{(4)} \ldots \cap S_K^{(K+1)}) + \ldots + \\ & \mathbb{P}(\bar{S}_1^{(1)} \cap \bar{S}_1^{(2)} \cap S_1^{(3)} \cap S_2^{(4)} \cap \ldots \cap S_K^{(K+2)}) + \\ & \mathbb{P}(\bar{S}_1^{(1)} \cap S_1^{(2)} \cap \bar{S}_2^{(3)} \cap (\bigcap_{k=2}^{K} S_k^{(k+2)})) + \ldots \end{aligned} \tag{19}$$

In the special case where all the link probabilities on a length K path are equal to p, we see that

$$\begin{aligned} \mathbb{P}(\bigcap_{k=1}^{K} S_k^{(k)}) &= p^K \\ \mathbb{P}(\bar{S}_1^{(1)} \cap (\bigcap_{k=1}^{K} S_k^{(k+1)})) &= (1-p)p^K \\ &\vdots \end{aligned} \tag{20}$$

In effect, if we let $r = t_d - K$, we want the probability that there are $i = 0, \ldots, r$ link failures over t_d independent trials. This is given by the cumulative distribution function of the negative binomial distribution [26], and we get a closed form expression for $p_{net}^{(t_d)}$

$$p_{net}^{(t_d)} = p^K \sum_{i=0}^{r} \binom{K+i-1}{i} (1-p)^i \quad . \tag{21}$$

APPENDIX B
PROOFS OF THEOREMS 2.1 AND 3.1

The proofs of Theorems 2.1 and 3.1 rely heavily on the following theorem:

Theorem B.1 (ρ_* determines convergence rate of $\mathbf{p}^{(t)}$): Let $P \in [0,1]^{N \times N}$ be a column stochastic matrix (meaning all the entries in the matrix are nonnegative and all the columns sum to 1) with $\lim_{k\to\infty} P^k \mathbf{p} = \mathbf{e}^{[b]}$ for all probability vectors $\mathbf{p} \in [0,1]^N, \sum_i \mathbf{p}_i = 1$. Here, $\mathbf{e}^{[b]}$ is an elementary vector with the b-th element equal to 1 and all other elements equal to 0. Let $\rho_* = \max\{|\lambda| : \lambda \text{ is an eigenvalue of } P \text{ and } |\lambda| < 1\}$. Then,

$$\left\| P^k \mathbf{p} - \mathbf{e}^{[b]} \right\|_1 \leq C k^{J-1} (\rho_*)^{k-J+1}, \quad \forall k \in \mathbb{Z}_+ \tag{22}$$

where C is a constant dependent on $\mathbf{p}$, and $J \in \mathbb{Z}_+$ is the size of the largest Jordan block of P. ■

The proof of this theorem is given later in Appendix B-C.

A. Proof of Theorem 2.1

Proof:

$$\begin{aligned} \left\| (P^{(\underline{T})})^k \mathbf{p}^{(0)} - \mathbf{e}^{[b]} \right\|_1 &= \left(\sum_{j \neq b} \left| \mathbf{p}_j^{(Tk)} \right| \right) + \left| \mathbf{p}_b^{(Tk)} - 1 \right| \\ &\overset{(a)}{=} \sum_{j \neq b} \left| \mathbf{p}_j^{(Tk)} \right| + \left| -\sum_{j \neq b} \mathbf{p}_j^{(Tk)} \right| \\ &\overset{(b)}{=} 2 \sum_{j \neq b} \mathbf{p}_j^{(Tk)} \end{aligned}$$

where step (a) uses the relationship $\mathbf{p}_b^{(t_d)} = 1 - \sum_{j \neq b} \mathbf{p}_j^{(t_d)}$ and step (b) uses the fact that the $\mathbf{p}_j^{(Tk)}$ are nonnegative.

By applying Theorem B.1 to $P^{(\underline{T})}$ we see that

$$\left\| (P^{(\underline{T})})^k \mathbf{p}^{(0)} - \mathbf{e}^{[b]} \right\|_1 \leq C k^{J-1} (\rho_*)^{k-J+1}, \quad \forall k \in \mathbb{Z}_+ \quad .$$

If we let $k = \lfloor \frac{t_d}{T} \rfloor$, we can combine the steps above to get

$$\begin{aligned} p_{net}^{(t_d)} = \mathbf{p}_b^{(t_d)} &\overset{(c)}{\geq} \mathbf{p}_b^{(Tk)} = 1 - \sum_{j \neq b} \mathbf{p}_j^{(Tk)} \\ &= 1 - \frac{1}{2} \left\| (P^{(\underline{T})})^k \mathbf{p}^{(0)} - \mathbf{e}^{[b]} \right\|_1 \\ &\geq 1 - \frac{1}{2} C k^{J-1} (\rho_*)^{k-J+1} \end{aligned}$$

where step (c) comes from the fact that b is an absorbing state in the Markov chain.

To summarize,

$$p_{net}^{(t_d)} \geq 1 - C k^{J-1} (\rho_*)^{k-J+1}, \quad k = \left\lfloor \frac{t_d}{T} \right\rfloor \tag{23}$$

for some constant C dependent on the initial distribution $\mathbf{p}^{(0)}$ and $J \in \mathbb{Z}_+$ the size of the largest Jordan block of $P^{(\underline{T})}$. Therefore, $p_{net}^{(t_d)}$ converges to 1 exponentially with a rate ρ_*. ■

B. Proof of Theorem 3.1

The steps in this proof are similar to the steps in the proof of Theorem 2.1.

Proof:

$$\begin{aligned}\left\|P^K\mathbf{p}^{(0)}-\mathbf{e}^{[\omega]}\right\|_1 &= \left(\sum_{j\neq\omega}\left|\mathbf{p}_j^{(K)}\right|\right)+\left|\mathbf{p}_\omega^{(K)}-1\right| \\ &= \sum_{j\neq\omega}\left|\mathbf{p}_j^{(K)}\right|+\left|-\sum_{j\neq\omega}\mathbf{p}_j^{(K)}\right| \\ &= 2\sum_{j\neq\omega}\mathbf{p}_j^{(K)}\end{aligned}$$

By applying Theorem B.1 to P we see that

$$\left\|P^K\mathbf{p}^{(0)}-\mathbf{e}^{[\omega]}\right\|_1 \leq CK^{J-1}(\rho_*)^{K-J+1}, \quad \forall K\in\mathbb{Z}_+ \quad .$$

Letting $t_d = KN_{stage}$ and combining the steps above, we get

$$\begin{aligned}p_{net}^{(t_d)} &= 1-\mathbf{p}_\omega^{(K)}=\sum_{j\neq\omega}\mathbf{p}_j^{(K)} \\ &= \frac{1}{2}\left\|P^K\mathbf{p}^{(0)}-\mathbf{e}^{[\omega]}\right\|_1 \\ &\leq \frac{1}{2}CK^{J-1}(\rho_*)^{K-J+1}\end{aligned}$$

To summarize,

$$p_{net}^{(t_d)} \leq CK^{J-1}(\rho_*)^{K-J+1}, \quad t_d = KN_{stage} \qquad (24)$$

for some constant C dependent on the initial distribution $\mathbf{p}^{(0)}$ and $J\in\mathbb{Z}_+$ the size of the largest Jordan block of P. Therefore, $p_{net}^{(t_d)}$ converges to 0 exponentially with a rate ρ_*. ■

C. Proof of Theorem B.1

1) Statement of Theorems and Lemmas Used in Proof: First, we state some theorems and definitions used in the proof, with the notation modified from their original sources to stay consistent with the notation used throughout this paper. Theorems B.2 and B.3 are not used explicitly in the proof, but are stated for the reader to better grasp Theorem B.4.

Theorem B.2 (Theorem 5.6.9 from [23]): If $\|\cdot\|$ is any matrix norm and if $A\in\mathbb{C}^{N\times N}$, then $\rho(A)\leq\|A\|$, where $\rho(A)\triangleq\max\{|\lambda| : \lambda \text{ is an eigenvalue of A}\}$ is the spectral radius of A. ■

Theorem B.3 (Spectral Radius of a Stochastic Matrix): The spectral radius (magnitude of the maximum eigenvalue) of a column stochastic matrix P is 1. ■

Proof: A proof of this can be be found in [27], and is reproduced here.

Since P is a column stochastic matrix, $\sum_{i=1}^N P_{ij}=1$ for all j. This means

$$\|P\|_1 \triangleq \max_j\sum_{i=1}^N|P_{ij}| = \max_j\sum_{i=1}^N P_{ij}=1$$

where $\|\cdot\|_1$ is the *maximum column sum norm*, and the first equality holds because $P_{ij}\geq 0$ for all i and j. Combining this with Theorem B.2, we see that $\rho(P)\leq 1$. ■

Periodic Markov Chains, from [22] A Markov chain is *periodic* if its states can be grouped in $d>1$ disjoint subsets $S_1,\ldots,S_d$ so that

$$\begin{array}{l}\text{if } i\in S_k \text{ and } p_{ij}>0, \\ \text{then } \begin{cases} j\in S_{k+1}, & \text{if } k=1,\ldots,d-1 \\ j\in S_1, & \text{if } k=d \end{cases}\end{array} .$$

A Markov chain is *aperiodic* if it is not periodic. ■

Decomposable Markov Chains, from [28] A Markov chain is *decomposable* if the state space $\mathcal{S}$ contains two non-empty disjoint subsets $\mathcal{S}_1$ and $\mathcal{S}_2$ which are closed, i.e. such that the probability that $i\in\mathcal{S}_1$ transitions to another node in $\mathcal{S}_1$ is 1 and the probability that $j\in\mathcal{S}_2$ transitions to another node in $\mathcal{S}_2$ is 1. ■

For the theorem below from Rosenthal, let λ_0 = 1 (the trivial eigenvalue of P) and $\rho_* = \max_{1\leq j\leq n-1}|\lambda_j|$, the largest absolute value of the nontrivial eigenvalues of P. From the theorem, we can also say that $\rho_* = \max\{|\lambda| : \lambda \text{ is an eigenvalue of } P \text{ and } |\lambda|<1\}$, which is used in the statement of the theorems of this paper. Other papers often refer to ρ_* as the second largest eigenvalue of the transition probability matrix.

Theorem B.4 (Fact 4 from [28]): A finite Markov chain satisfies $\rho_*<1$ if and only if it is both indecomposable and aperiodic. ■

For the theorem below from Rosenthal, let the *total variation distance* between probability measures v_1 and v_2 be defined as $\|v_1-v_2\|_{var}\triangleq\sup_{A\subset\mathcal{S}}|v_1(A)-v_2(A)|$. Then, if $\mathcal{S}$ is finite, $\|v_1-v_2\|_{var}=\frac{1}{2}\sum_{i\in\mathcal{S}}|v_1(i)-v_2(i)|$.

Theorem B.5 (Part of Fact 3 from [28]): Suppose P satisfies $\rho_*<1$ and the state space $\mathcal{S}$ is finite. Then, there is a unique stationary distribution π on $\mathcal{S}$ and, given an initial distribution $\mathbf{p}^{(0)}$ and point $i\in\mathcal{S}$, there is a constant $C_i>0$ such that

$$|\mathbf{p}_i^{(k)}-\pi_i| \leq C_i k^{J-1}(\rho_*)^{k-J+1}$$

where J is the size of the largest Jordan block of P. It follows immediately that

$$\|\mathbf{p}^{(k)}-\pi\|_{var} \leq Ck^{J-1}(\rho_*)^{k-J+1} \qquad (25)$$

where $C=\frac{1}{2}\sum C_i$. In particular, if P is diagonalizable (so that $J=1$) then

$$\begin{aligned}\|\mathbf{p}_i^{(k)}-\pi_i\|_{var} &\leq \sum_{m=1}^{n-1}|a_m\mathbf{v}_m(i)||\lambda_m|^k \\ &\leq \left(\sum_{m=1}^{n-1}|a_m\mathbf{v}_m(i)|\right)(\rho_*)^k\end{aligned}$$

where $\mathbf{v}_0,\ldots,\mathbf{v}_{n-1}$ are a basis of right eigenvectors corresponding to $\lambda_0,\ldots,\lambda_{n-1}$ respectively, and where a_m are the (unique) complex coefficients satisfying

$$\mathbf{p}^{(0)}=a_0\mathbf{v}_0+a_1\mathbf{v}_1+\ldots+a_{n-1}\mathbf{v}_{n-1} \quad .$$

Here, $\mathbf{v}_m(i)$ denotes the i-th coordinate of the vector $\mathbf{v}_m$. ■

For finite $\mathcal{S}$, we can relate the 1-norm to the total variation distance by

$$\|v_1 - v_2\|_{var} = \frac{1}{2}\sum_i |v_1(i) - v_2(i)| = \frac{1}{2}\|v_1 - v_2\|_1 \quad . \tag{26}$$

This means that Equation 25 can be restated as

$$\|\mathbf{p}^{(k)} - \pi\|_1 \leq Ck^{J-1}(\rho_*)^{k-J+1} \tag{27}$$

where $C = \sum C_i$.

2) Proof of Theorem B.1:

Proof: A column stochastic matrix $P \in [0,1]^{N\times N}$ with $\lim_{k\to\infty} P^k\mathbf{p} = \mathbf{e}^{[b]}$ for all probability vectors $\mathbf{p} \in [0,1]^N, \sum_i \mathbf{p}_i = 1$, describes the transition probability matrix for a Markov chain that is both indecomposable and aperiodic. The Markov chain is not decomposable because a decomposable Markov chain has more than one stationary distribution, whereas the Markov chain described by P has a unique stationary distribution $\mathbf{e}^{[b]}$. For instance, a decomposable Markov chain would have a stationary probability distribution with nonzero entries over only the states in $\mathcal{S}_1$, and another stationary probability distribution with nonzero entries over only the states in $\mathcal{S}_2$. The Markov chain described by P is aperiodic because all probability distributions converge to a unique stationary distribution, meaning that there is no distribution that transitions in a periodic manner over time.

Since the Markov chain described by P is both indecomposable and aperiodic, we can apply Theorem B.4 and Theorem B.5 to get the desired result, where $\mathbf{p}^{(k)}$ corresponds to $P^k\mathbf{p}$ and π corresponds to $\mathbf{e}^{[b]}$. ■

3) Discussion: The proof of Theorem B.1 appears to rely heavily on the assumption $\lim_{k\to\infty} P^k\mathbf{p} = \mathbf{e}^{[b]}$ for all probability vectors $\mathbf{p} \in [0,1]^N, \sum_i \mathbf{p}_i = 1$. For the MTMC model, this corresponds to modeling a routing topology with a unique sink/destination node where all packets are eventually routed to this sink. If we wish to apply this theorem to mesh networks with multiple collection points, as mentioned in Section II-C, we need to make some simple modifications to the Markov chain model.

First, we would combine the states $i \in \mathcal{B}$ representing the collection/destination nodes into one state $i_\mathcal{B}$ in the MTMC model. The transition probabilities to this new state $i_\mathcal{B}$ would be $p_{ii_\mathcal{B}} = \sum_{j\in\mathcal{B}} p_{ij}$ while the transition probabilities $p_{i_\mathcal{B}j}$ out of $i_\mathcal{B}$ would be

$$p_{i_\mathcal{B}j} = \begin{cases} 1 & : \quad j = i_\mathcal{B} \\ 0 & : \quad j \neq i_\mathcal{B} \end{cases}$$

meaning $i_\mathcal{B}$ is a recurrent state. We can now apply Theorem B.1 to this new Markov chain model to show that the model converges to $i_\mathcal{B}$ at rate ρ_*. This means that the packet will eventually reach one of the collection/destination nodes at rate ρ_*, although the packet arrival probability distribution over the nodes in $\mathcal{B}$ may depend on which node originally sent the packet.

APPENDIX C
FULL SIZED TRANSITION MATRIX FOR FIGURE 5

The 8×8 matrix is broken up into two "lines" so it fits in one column at a larger font size and can be easily read.

$$\begin{bmatrix}
1 & \bar{p}^2 & \bar{p}^3 & \bar{p}^5 & \bar{p}^2 & \dots & \dots & \bar{p}^4 & \bar{p}^5 & \bar{p}^7 \\
0 & p\bar{p} & p\bar{p}^2 & (1-\bar{p}^2)\bar{p}^3 & 0 & \dots & \dots & p\bar{p}^3 & p\bar{p}^4 & (1-\bar{p}^2)\bar{p}^5 \\
0 & p\bar{p} & p\bar{p}^2 & (1-\bar{p}^2)\bar{p}^3 & p\bar{p} & \dots & \dots & (1-\bar{p}^2)\bar{p}^2 & (1-\bar{p}^2)\bar{p}^3 & (1-\bar{p}^3)\bar{p}^4 \\
0 & p^2 & p^2\bar{p} & (1-\bar{p}^2)^2\bar{p} & 0 & \dots & \dots & p(1-\bar{p}^2)\bar{p} & p(1-\bar{p}^2)\bar{p}^2 & (1-\bar{p}^2)(1-\bar{p}^3)\bar{p}^2 \\
0 & 0 & p\bar{p}^2 & p\bar{p}^4 & p\bar{p} & \dots & \dots & p\bar{p}^3 & (1-\bar{p}^2)\bar{p}^3 & (1-\bar{p}^2)\bar{p}^5 \\
0 & 0 & p^2\bar{p} & p(1-\bar{p}^2)\bar{p}^2 & 0 & \dots & \dots & p^2\bar{p}^2 & p(1-\bar{p}^2)\bar{p}^2 & (1-\bar{p}^2)^2\bar{p}^3 \\
0 & 0 & p^2\bar{p} & p(1-\bar{p}^2)\bar{p}^2 & p^2 & \dots & \dots & p(1-\bar{p}^2)\bar{p} & (1-\bar{p}^2)^2\bar{p} & (1-\bar{p}^2)(1-\bar{p}^3)\bar{p}^2 \\
0 & 0 & p^3 & p(1-\bar{p}^2)^2 & 0 & \dots & \dots & p^2(1-\bar{p}^2) & p(1-\bar{p}^2)^2 & (1-\bar{p}^2)^2(1-\bar{p}^3)
\end{bmatrix} \tag{28}$$

APPENDIX D
COMPARISON OF UPD WITH DIFFERENT SCHEDULES

The left graph of Figure 19 shows that the end-to-end connectivity function $p_{net}^{(t_d)}$ can differ significantly even for seemingly small changes in the routing schedule. Consider the *packed* routing schedule in the left diagram of Figure 18, which is "efficient" in the sense that all nodes except for the nodes next to the source and sink transmit or receive at every time slot. Then consider the *unpacked* schedule in the right diagram of Figure 18, which is not efficient in this sense because on every odd time slot, one node in each column is not transmitting or receiving. This "inefficiency" is enough that to reach or exceed some values of p_{net}, the unpacked schedule requires as much as one extra superframe (6 time slots) of latency as the packed schedule. The difference in performance may be even greater if we change the order of the time slots in a superframe, as illustrated on the left graph Figure 19 by the *scrambled unpacked* schedule described in the caption of Figure 18. Of course, the performance of one routing schedule with respect to another depends on the link probabilities in the network. For instance, at extremely low link probabilities such as $p_l = 0.2$ the scrambled unpacked schedule has performance close to that of the unpacked schedule, as shown on the right graph of Figure 19 (and in fact at certain times is slightly better, though it is hard to see on the graph).

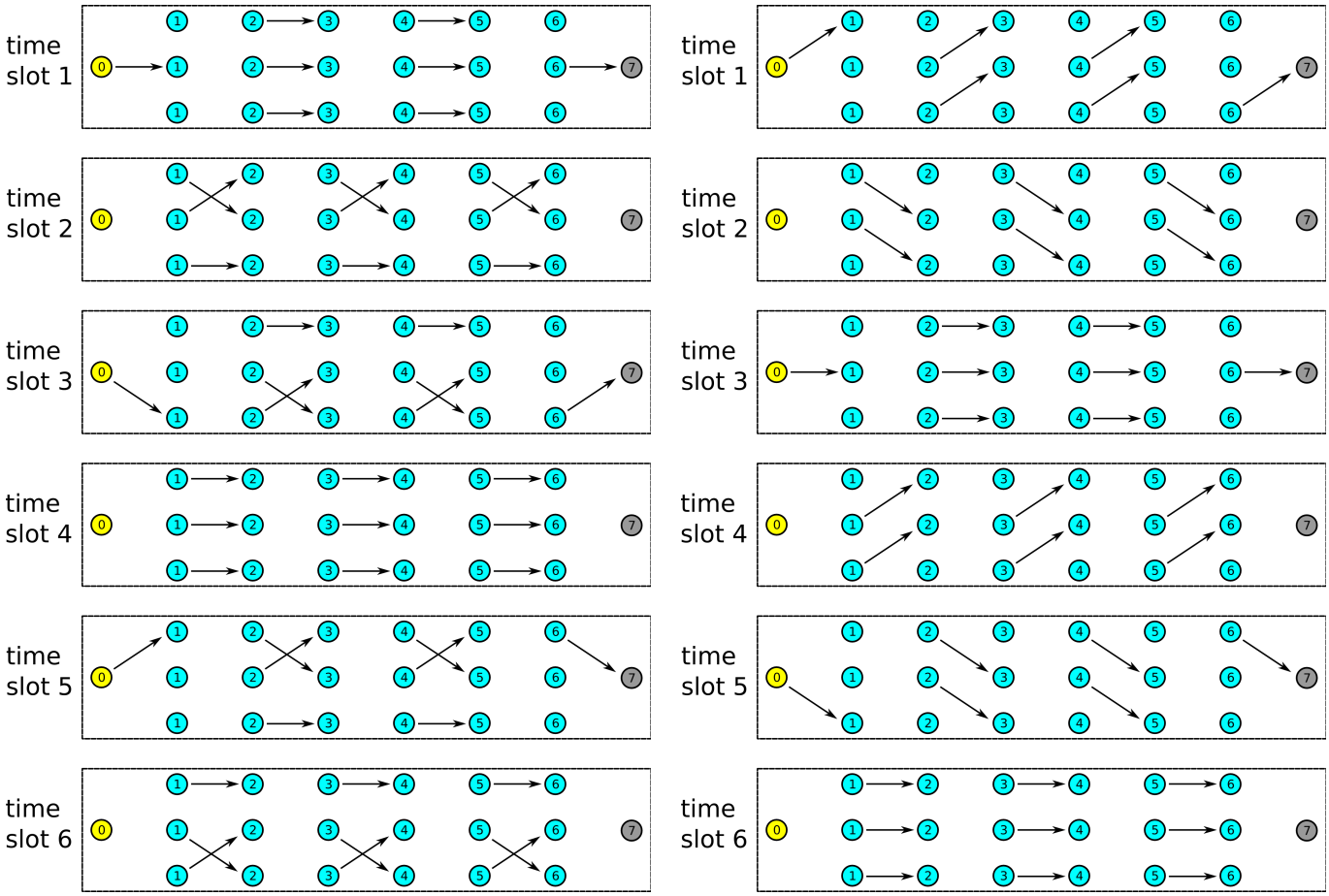

Fig. 18. Different UPD schedules on paths of width 3: (left) *packed schedule*, where all nodes except those neighboring the source and destination transmit or receive at each time slot; (right) *unpacked schedule*, where some nodes are idle at each time slot. The *scrambled unpacked schedule* is the unpacked schedule with the time slots permuted by $(1, 3, 5, 2, 4, 6)$.

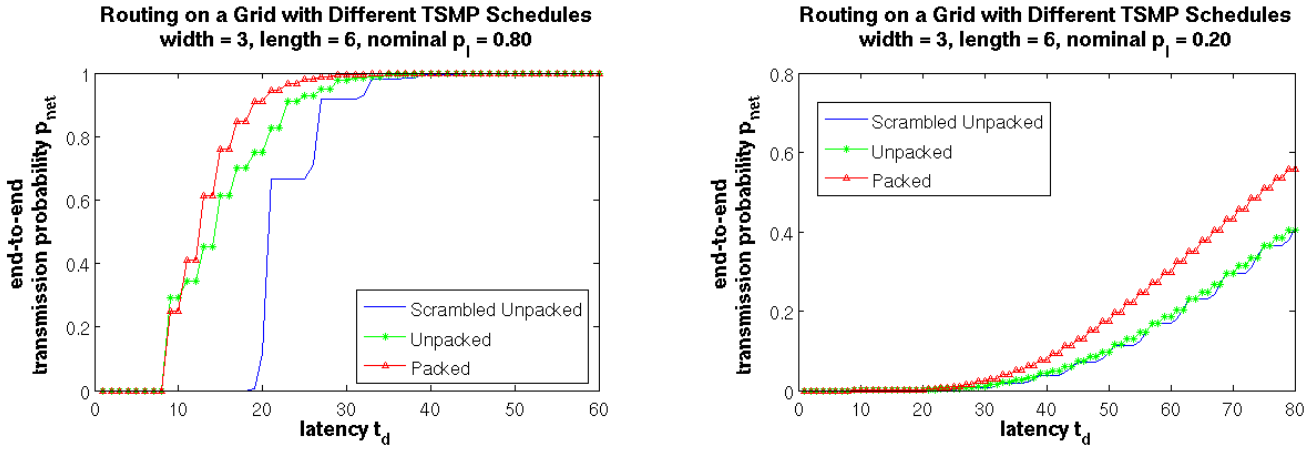

Fig. 19. End-to-end connectivity function $p_{net}^{(t_d)}$ varies greatly depending on the choice of UPD routing schedules. The routing schedules are depicted in Figure 18.