Optimal Throughput-Delay Trade-off in Wireless Networks – Part II: Constant-Size Packets

Abbas El Gamal, James Mammen, Balaji Prabhakar and Devavrat Shah

Abstract

In previous work (2004), we characterized the optimal throughput-delay trade-off in static wireless networks as $D(n) = \Theta(nT(n))$, where $D(n)$ and $T(n)$ are the average packet delay and throughput in a network of n nodes, respectively. While this trade-off captured the essential network dynamics, packets needed to scale down with the network size. In this "fluid model", no buffers were required. Due to this packet scaling, $D(n)$ did not correspond to the average delay per bit. That led to the question whether the trade-off remains the same when the packet size is kept constant, which necessitates buffers and packet scheduling in the network.

In this paper, we answer this question in the affirmative by showing that the optimal throughput-delay trade-off is still $D(n) = \Theta(nT(n))$, where now $D(n)$ is the average delay per bit. Packets of constant size necessitate the use of buffers in the network, which in turn requires scheduling packet transmissions in a discrete-time queueing network and analyzing the corresponding delay. Our method consists of deriving packet schedules in the discrete-time network by looking at a corresponding continuous-time network and then analyzing the delay induced in the actual discrete network using results from queueing theory for continuous-time networks.

I. Introduction

Gupta and Kumar, in their seminal work [4], introduced a random network model for studying throughput scaling in a static wireless network, i.e., when the nodes do not move. They showed that the throughput per source-destination pair scales as $\Theta\left(1/\sqrt{n\log n}\right)$. They implicitly used a fluid model and later work by Kulkarni and Viswanath [6] consolidated the result with an explicit constant packet size model.

In previous work [1], we studied the throughput-delay trade-off in wireless networks and a more complete treatment is provided in part I [2] of this work. The optimal throughput-delay trade-off was established to be $D(n) = \Theta(nT(n))$, which is graphically presented in Figure 1. In this work, packet size needed to scale down with the number of nodes n in the network. This led to a fluid model for transmitting packets and allowed us to obtain the essential trade-off by skirting the issue of buffering and the resultant queueing delay at the nodes. The delay that was considered in [1] was the average packet delay and since the packet size was allowed to scale down with n, it did not correspond to the average delay per bit. This paper investigates the throughput-delay trade-off when the packet size remains constant, i.e., does not scale down with n. This is an important question, since in real networks, packet sizes do not change when more nodes are added to the network. Note that with the additional constraint that the packet size remains constant, the throughput-delay trade-off can be no better than that in the fluid model. However, a priori, it is not clear whether the same throughput-delay trade-off as in the fluid case can be achieved, since now, routing packets through the network also involves the additional task of scheduling in the network. In this paper, we extend our previous work to the case of wireless networks with buffers and constant-size packets and show that the optimal trade-off is still $D(n) = \Theta(nT(n))$ (as shown in Figure 1) where now $D(n)$ is the average delay per bit.

The main contribution of this paper is to determine the exact order of delay by coupling the evolution of a discrete-time queueing network with that of a continuous-time queueing network. This provides both a packet scheduling policy (see item 6 of Policy Σ_n in Section II) and a method for analyzing the delay. Packets in a wireless network have fixed routes depending on the source-destination pair to which they belong. The entire wireless network then corresponds to a discrete-time, open queueing network with general customer routes, in the terminology of queueing theory (e.g. see [5], [8]). In the case of continuous-time queueing networks, these are also known as Kelly or BCMP networks and the equilibrium distribution is known to have a product form. Since packet size is a constant and does not have an Exponential distribution, we use Preemptive LIFO to obtain a symmetric queue,

The authors are with the departments of EE and CS at Stanford University

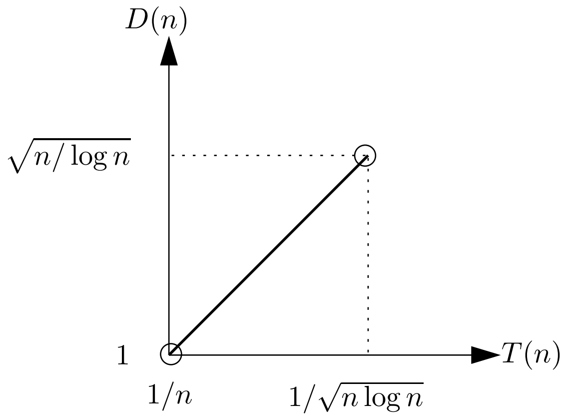

Fig. 1. Throughput-delay scaling trade-off in the static random network model. The scales of the axes are in terms of orders in n.

in order to have a product form equilibrium distribution. Then based on packet arrival times in a continuous-time queueing network with a Preemptive LIFO queue management at each server, we derive a scheduling policy for the wireless network. Finally, using product form equilibrium results for continuous-time networks, we determine the exact order of queueing delay in the discrete-time wireless network.

A. Model and Definitions

For the sake of completeness, we repeat the models and definitions already presented in [2]. Refer to [2] for a more complete explanation of the model.

Definition 1 (Static random network model): The static random network consists of a unit torus in which n nodes are distributed uniformly at random. These n nodes are split into $n/2$ distinct source-destination (S-D) pairs at random. Time is slotted for packetized transmission. For simplicity, we assume that the time-slots are of unit length.

Definition 2 (Model for successful transmission): Under our *Relaxed Protocol* model, a transmission from node i to node j in a time-slot is successful if for any other node k that is transmitting simultaneously,

$$d(k, j) \geq (1 + \Delta)d(i, j) \quad \text{for} \quad \Delta > 0$$

where $d(i, j)$ is the distance between nodes i and j. During a successful transmission, nodes send data at a constant rate of W bits per second.
With time-slots of unit length, this means that the size of packets transmitted in each slot is W bits.

Definition 3 (Scheme): A scheme Π, for a random network, is a sequence of communication policies, (Π_n), where policy Π_n determines how communication occurs in a network of n nodes.

Definition 4 (Throughput of a scheme): Let $B_{\Pi_n}(i, t)$ be the number of bits of S-D pair $i, 1 \leq i \leq n/2$, transferred in t time-slots under policy Π_n. Note that this could be a random quantity for a given realization of the network. Scheme Π is said to have throughput $T_\Pi(n)$ if there exists a sequence of sets $A_\Pi(n)$ such that

$$A_\Pi(n) = \left\{ \omega : \min_{1 \leq i \leq n/2} \liminf_{t \to \infty} \frac{1}{t} B_{\Pi_n}(i, t) \geq T_\Pi(n) \right\}$$

and $P\left(A_\Pi(n)\right) \to 1$ as $n \to \infty$.
We use the term *whp* (with high probability) to denote this.

Definition 5 (Delay of a scheme): The delay of a bit is the time it takes for the bit to reach its destination after it leaves the source. Let $D^i_{\Pi_n}(j)$ denote the delay of bit j of S-D pair i under policy Π_n, then the sample mean of delay for S-D pair i under is

$$\bar{D}^i_{\Pi_n} = \limsup_{k \to \infty} \frac{1}{k} \sum_{j=1}^{k} D^i_{\Pi_n}(j).$$

The delay for scheme Π is the average delay over all S-D pairs, i.e.,

$$D_\Pi(n) = \frac{2}{n} \sum_{i=1}^{n/2} \bar{D}^i_{\Pi_n}.$$

When an equilibrium distribution exists under policy Π_n, $D_\Pi(n)$ is equal to the expected value of delay under the equilibrium distribution when each S-D pair communicates at the same rate.

Definition 6 (Throughput-delay optimality): A pair $(T(n), D(n))$ is said to be Throughput-Delay (T-D) optimal if there exists a scheme Π with $T_\Pi(n) = \Theta(T(n))$ and $D_\Pi(n) = \Theta(D(n))$ and $\forall$ scheme Π' with $T_{\Pi'}(n) = \Omega(T(n))$, $D(\Pi')(n) = \Omega(D(n))$.

Definition 7 (Optimal througput-delay trade-off): The optimal throughput-delay trade-off consists of all the T-D optimal pairs.

Regarding delay, we would like to note that in the definition we used bit delay whereas in the schemes we present later, we refer to packet delay since communication is packetized. But since the packet size is constant, these quantities are of the same order. In this paper, we use c_i to denote constants that do not depend on n.

Our main result is as follows.

Theorem 1: The optimal throughput-delay trade-off in the static random network model is given by

$$T(n) = \Theta\left(D(n)/n\right),$$

for $T(n) = O\left(1/\sqrt{n \log n}\right)$.

The above result says that under a delay scaling constraint of $D(n)$ the optimal throughput scaling is $\Theta(D(n)/n)$. And this holds for $T(n) = O\left(1/\sqrt{n \log n}\right)$, that is, the entire range of achievable throughputs in the static random network model.

The rest of this paper is organized as follows. In Section II, we introduce Scheme Π and show that it achieves the throughput-delay trade-off stated in Theorem 1. Finally we present a converse that shows that no scheme can provide a better throughput-delay trade-off than Scheme Π, thus establishing Theorem 1.

II. Throughput-Delay Trade-off in Static Networks

Our trade-off scheme is a multi-hop, time-division-multiplexed (TDM), cellular scheme with square cells of area $a(n)$ so that the unit torus consists of $1/a(n)$ cells as shown in Figure 2. In the following analysis, we ignore the edge effects due to $1/a(n)$ not being a perfect square. Before presenting the trade-off scheme, we present three lemmas about the geometry of the n nodes on the torus divided into square cells of area $a(n)$. See [2] for the proofs.

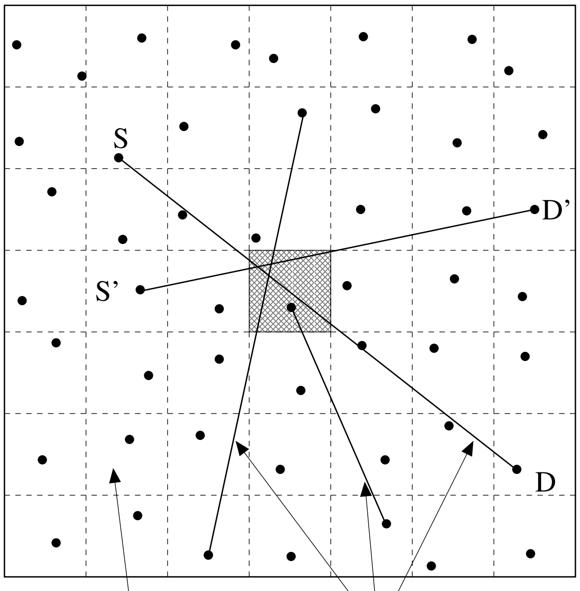

Fig. 2. The unit torus is divided into cells of size $a(n)$ for Scheme Π. The S-D lines passing through the shaded cell in the center are shown.

Lemma 1: If $a(n) \geq 2 \log n/n$, then each cell has at least one node *whp*.

We say that cell B *interferes* with another cell A if a transmission by a node in cell B can affect the success of a simultaneous transmission by a node in cell A.

Lemma 2: Under the Relaxed Protocol model, the number of cells that interfere with any given cell is bounded above by a constant c_1, independent of n.

We say that a cell is *active* in a time-slot if any of it nodes transmits in that time-slot. A consequence of Lemma 2 is that, there exists an interference-free schedule where each cell becomes active regularly, once in $1+c_1$ time-slots and no cell interferes with any other simultaneously transmitting cell.

Let the straight line connecting a source S to its destination D be called an S-D line.

Lemma 3: The number of S-D lines passing through each cell is $O\left(n\sqrt{a(n)}\right)$, *whp*.

The above lemma shows that the number of S-D lines passing through each cell is $\leq c_2 n\sqrt{a(n)}$ *whp*, for an appropriate choice of the constant c_2.

Now we are ready to describe Scheme Π, which is parameterized by the cell area $a(n)$ where $a(n) = \Omega(\log n/n)$ and $a(n) \leq 1$. Recall that by definition, Scheme Π is a sequence of communication policies (Π_n). For any particular realization of the random network with n nodes, policy Π_n differs based on the following two conditions.

Condition 1: No cell is empty.

Condition 2: The number of S-D lines through each cell is at most $c_2 n\sqrt{a(n)}$.

If both the above conditions above are satisfied then Π_n is the policy Σ_n, described below. Otherwise, Π_n is a time-division policy where each of the $n/2$ sources transmits directly to its destination in a round-robin fashion.

Policy Σ_n:

---

1) Divide the unit torus using a square grid into square cells, each of area $a(n)$ (see Figure 2).
2) Each node generates packets according to a Poisson process of rate $T(n) = \Theta\left(1/n\sqrt{a(n)}\right)$. The random network is a discrete-time system whereas the packet generation is a continuous-time process. So if a packet is generated at time t, it is available for transmission from time-slot $\lceil t \rceil$ onwards.
3) Each cell becomes active at a regular interval of $1 + c_1$ time-slots (see Lemma 2). Several cells which are sufficiently far apart become active simultaneously. Thus the scheme uses TDM between nearby cells.
4) A source S sends packets to its destination D by relaying or hopping along the adjacent cells lying on its S-D line as shown in Figure 2. Thus, in this scheme, direct transmission of packets is only between nodes in adjacent cells.
5) One of the nodes in a cell acts as a relay by maintaining a buffer for the packets of all the S-D lines passing through that cell. In each time-slot only one packet can be transmitted. However, a relay node may receive up to four packets from its adjacent cells before it gets a chance to relay them. Moreover multiple packets may be generated within the cell which will be available for transmission in the next time-slot. Hence a virtual queue is formed in each cell which consists of packets generated within the cell as well as the packets to be relayed through the cell.
6) When the cell becomes active, one packet from this queue (if not empty) is transmitted to an adjacent cell according to a Last-In-First-Out (LIFO) type of queue service policy. However, the arrival times considered by this policy are not the actual arrival times of the packets, but the arrival times that would occur in a continuous-time network with the same arrivals and a PL (Preemptive LIFO) queue management at each server. This is elaborated later in this section during the analysis of delay.

The point of trade-off at which Scheme Π operates is determined by the parameter $a(n)$ and the dependence is made precise in the following theorem.

Theorem 2: For $a(n) = \Omega(\log n/n)$,

$$T(n) = \Theta\left(1/n\sqrt{a(n)}\right) \quad \text{and} \quad D(n) = \Theta\left(1/\sqrt{a(n)}\right),$$

i.e., the throughput-delay trade-off achieved by Scheme Π is

$$T(n) = \Theta\left(D(n)/n\right).$$

Throughput analysis: If the time-division policy with direct transmission is used, then the throughput is $2W/n$ and delay of 1. But since it happens with a vanishingly low probability, as shown by Lemmas 1 and 3, the throughput and delay for Scheme Π are determined by that of policy Σ_n.

When policy Σ_n is used, since Condition 1 is satisfied, each cell has at least one node. This guarantees that each source can send data to its destination by hops along adjacent cells on its S-D line. From Lemma 2, it follows that each cell gets to transmit a packet every $1 + c_1$ time-slots, or equivalently, the cell throughput is $\Theta(1)$. The total traffic through each cell is that due to all the S-D lines passing through the cell, which is $O\left(n\sqrt{a(n)}\right)$ since Condition 2 is also satisfied. This suggests that

$$T(n) = \Theta\left(n\sqrt{a(n)}\right),$$

is achievable, if the average delay is finite.

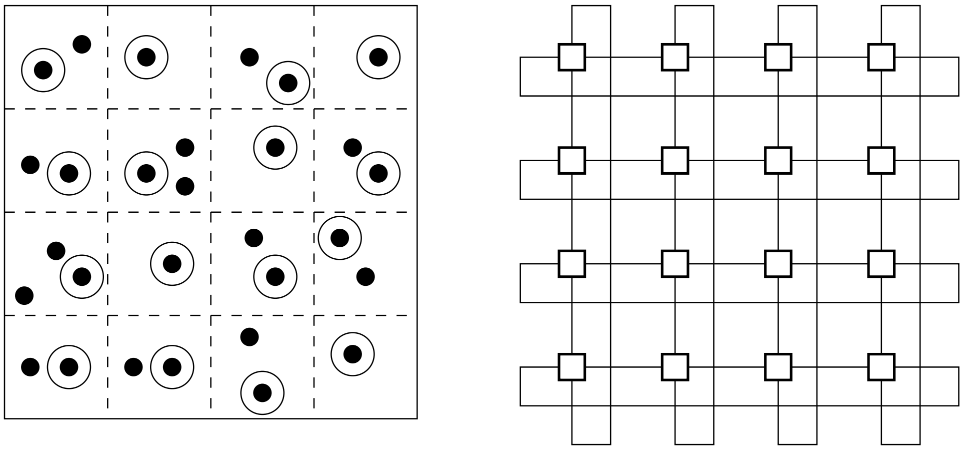

Fig. 3. The torus on the left with has 16 cells and each cell contains at least one node. The circled node in each cell acts as a relay. The corresponding queueing network of 16 servers, with each server corresponding to a cell in the wireless network, is shown on the right.

Delay Analysis: Next we analyze the average packet delay in the wireless network for Scheme Π when Conditions 1 and 2 are satisfied, i.e, when policy Σ_n is used. Dividing the unit torus into square cells of area $a(n)$ results in $1/a(n)$ cells. One of the nodes in each cell maintains a buffer and acts as a relay for all the S-D lines passing through that cell. These relay nodes are the circled nodes in Figure 3. The buffer in each cell corresponds to a queue and the cell itself corresponds to a server that can transmit one packet from this queue once in $1 + c_1$ time-slots. This is because each cell becomes active once in $1 + c_1$ time-slots as described earlier. Since Scheme Π restricts direct transmissions to be between adjacent cells, each cell can receive from or transmit to any four of its adjacent cells. This determines the connectivity between the servers so that the entire wireless network corresponds to a discrete-time queueing network of $1/a(n)$ servers, where each server is connected to four others as shown in Figure 3.

Note that the time-division-multiplexing between cells is such that in the c_1 slots before each cell becomes active again each of its neighbors becomes active exactly once. Hence we can ignore the effect of cells becoming active at regular intervals and instead consider a discrete-time network of queues $\mathcal{N}_D$ where D signifies the discrete time nature of this network. The actual delay in the wireless network would then be $1 + c_1$ times the delay in $\mathcal{N}_D$.

Queueing network $\mathcal{N}_D$: The discrete-time queueing network $\mathcal{N}_D$ consists of $1/a(n)$ servers, each of which can service one packet from its queue in a time-slot if it is not empty. Moreover, each server is connected to four others as explained above. In the wireless network, packets travel from their sources to their destinations by hops along adjacent cells on their S-D lines. Thus the route of a packet depends on the S-D pair to which it belongs. This means that in $\mathcal{N}_D$ there are $n/2$ customer routes corresponding to the $n/2$ S-D pairs. Recall that packets arrive in the wireless network at the sources according to independent Poisson processes of rate $T(n)$. These correspond to exogenous arrivals at the queues in $\mathcal{N}_D$. The remaining arrivals at the queues are due to the departures from other queues. In the terminology of queueing theory, $\mathcal{N}_D$ is a discrete-time, open network of queues with general customer routes (see Chapter 6.6 of [8]).

Delay analysis for such discrete-time networks with general customer routes is not known, which prevents us from using a simple First-In-First-Out (FIFO) order of service in $\mathcal{N}_D$. We leverage results known about continuous-time networks to obtain the queue management policy for $\mathcal{N}_D$ in such a way that the average delay can be computed.

Queueing network $\mathcal{N}_C$: Consider a continuous-time open network of $1/a(n)$ servers having the same connectivity structure as $\mathcal{N}_D$ and the same $n/2$ customer routes (see Figure 3). Let this network be called $\mathcal{N}_C$. Further, let the exogenous arrivals in both the networks $\mathcal{N}_C$ and $\mathcal{N}_D$ are the same. And let the service requirement of each packet at each server be deterministically equal to unit time. From the description until now, it is clear that $\mathcal{N}_C$ is the continuous-time analog of the discrete-time network $\mathcal{N}_D$. A Preemptive LIFO (PL) queue management is used at each server in $\mathcal{N}_C$ (see Chapter 6.8 of [8] for more details).

The queue size distribution for the continuous time network $\mathcal{N}_C$ with PL queue management at each server has a product form in equilibrium as shown in [5] (see Theorems 3.7 and 3.8 of Chapter 3) provided that the following two conditions are satisfied. First, the service time distribution should be either phase-type or the limit of a sequence of phase-type distributions. In our case the service time is constant and equal to 1. The second condition is that, the total traffic at each server is less than its capacity, which is one in our case.

Consider the sum of n exponential random variables each with mean $1/n$. This sum has a phase-type distribution and in the limit as n tends to infinity, its distribution converges to that of a constant random variable. Thus the first condition is satisfied.

In the wireless network the number of S-D lines passing through each cell is $O\left(n\sqrt{a(n)}\right)$ and the arrival process for each S-D pair is an independent Poisson process with rate $T(n) = \Theta\left(1/n\sqrt{a(n)}\right)$. Therefore an appropriate choice of constants guarantees that the total traffic at each server is less than 1, its service capacity, due to Condition 2 being satisfied.

Using the product form for the queue size distribution in equilibrium, it follows that the average queue size at a queue with total traffic $\lambda < 1$ and unit mean service is of the form $c_3\lambda/(1-\lambda)$ where c_3 is some constant. By Little's law this implies that the average delay at each server is bounded above by a constant independent of n. We summarize the above discussion in the lemma below.

Lemma 4: For the continuous-time open network $\mathcal{N}_C$ with $n/2$ customer routes as described above the average delay at each server is bounded above by a constant independent of n.

Packet Scheduling in $\mathcal{N}_D$ using $\mathcal{N}_C$: However we cannot use this PL policy in the discrete time network $\mathcal{N}_D$ because of the following reasons:

1) Due to the discrete time nature of the network $\mathcal{N}_D$, a packet that is generated at time t becomes eligible for service (i.e. next hop transmission) only at time $\lceil t \rceil$.
2) A complete packet has to be transmitted in a time-slot, i.e. fractions of the packets cannot be transmitted. This means that a preemptive type of service like PL is not allowed.

To address these problems for $\mathcal{N}_D$, we present a centralized scheduling policy derived from emulating in parallel, the continuous-time network $\mathcal{N}_C$ with PL queue management at each server. The exogenous arrivals in both $\mathcal{N}_C$ and $\mathcal{N}_D$ are the same. Let a packet arrive in $\mathcal{N}_C$ at some server at time a_C and in $\mathcal{N}_D$ at the same server at time a_D. Then it is served in $\mathcal{N}_D$ using a LIFO policy with the arrival time considered to be $\lceil a_C \rceil$ instead of a_D.

Clearly such a policy can be implemented if and only if $a_D \le \lceil a_C \rceil$ for every packet at each server, i.e., each packet arrives before its scheduled departure time. Let d_C and d_D be the departure times of a packet from some server in $\mathcal{N}_C$ and $\mathcal{N}_D$ respectively. Then this is the same as saying that $d_D \le \lceil d_C \rceil$ for each packet in every busy cycle of each server in $\mathcal{N}_C$. In what follows, we show that for all packets in any busy cycle of any server, the departures in $\mathcal{N}_D$ occur at or before the departures in $\mathcal{N}_C$.

Lemma 5: Let a packet depart in $\mathcal{N}_C$ from some server at time d_C and in $\mathcal{N}_D$ at time d_D, then $d_D \le \lceil d_C \rceil$.

Proof: Fix a server and a particular busy cycle of $\mathcal{N}_C$. Let it consist of packets numbered $1, \ldots, k$ with arrivals at times $a_1 \le \ldots \le a_k$ and departures at times $d_1, \ldots, d_k$. Let the arrival times of these packets in $\mathcal{N}_D$ be $A_1, \ldots, A_k$ and departures be at times $D_1, \ldots, D_k$. By assuming that $A_i \le \lceil a_i \rceil$ for $i = 1, \ldots, k$, we need to show that $D_i \le \lceil d_i \rceil$ for $i = 1, \ldots, k$.

Clearly this holds for $k = 1$ since $D_1 = \lceil A_1 \rceil + 1 \le \lceil a_i \rceil + 1 = \lceil d_1 \rceil$. Now suppose it holds for all busy cycles of length k and consider any busy cycle of $k+1$ packets.

If $\lceil a_1 \rceil < \lceil a_2 \rceil$, then because of the LIFO policy in $\mathcal{N}_D$ based on times a_i, we have $D_1 = \lceil a_1 \rceil + 1 \le \lceil a_1 \rceil + k + 1 = \lceil d_1 \rceil$. The last equality holds since in $\mathcal{N}_C$, the PL service policy dictates that the first packet of the busy cycle is the last to depart. And the remaining packets would have departures times as for a busy cycle of length k.

Otherwise if $\lceil a_1 \rceil = \lceil a_2 \rceil$ then the LIFO policy in $\mathcal{N}_D$ based on arrival times a_i results in $D_1 = \lceil a_1 \rceil + k + 1 = \lceil d_1 \rceil$ and the packets numbered $2, \ldots, k$ depart exactly as if they belong to a busy cycle of length k. This completes the proof by induction. ■

Thus we have shown that it is possible to use LIFO in $\mathcal{N}_D$ based on the arrival times in $\mathcal{N}_C$ instead of the actual arrival times in $\mathcal{N}_D$. We are now ready to prove Theorem 2.

Proof: (of Theorem 2) Packets reach their destination with finite average delay, which shows that the throughput is just the rate at which each source sends its data. This proves that the throughput $T(n) = \Theta\left(1/n\sqrt{a(n)}\right)$.

Next we compute the average packet delay $D(n)$. Lemma 5 also holds for the final departure of each packet from the network. Therefore if D_D^i is the delay of a packet of route i in $\mathcal{N}_D$ (i.e. S-D pair i in the wireless network) and D_C^i is the delay of the corresponding packet in $\mathcal{N}_C$ then $D_D^i \leq D_C^i + 1$. Hence taking expectations it follows that

$$E[D_D^i] \leq E[D_C^i] + 1, \quad 1 \leq i \leq n/2.$$

Therefore delay averaged over all $n/2$ routes is given by

$$D(n) = \frac{2}{n} \sum_{i=1}^{n/2} E[D_D^i] \leq \frac{2}{n} \sum_{i=1}^{n/2} E[D_C^i] + 1. \tag{1}$$

Since each hop in the wireless network covers a distance of $\Theta\left(\sqrt{a(n)}\right)$, the number of hops per packet for S-D pair i is $\Theta\left(d_i/\sqrt{a(n)}\right)$ where d_i is the length of S-D line i. Now D_C^i is the delay for a packet of route i, which is equal to the sum of the delays along all queues on its route. But from Lemma 4, the average delay at each server is bounded above by some constant independent of n. Therefore from (1), we obtain that

$$D(n) \leq \frac{2}{n} \sum_{i=1}^{n/2} c_2 \frac{E[d_i]}{\sqrt{a(n)}} + 1 = \Theta\left(1/\sqrt{a(n)}\right)$$

since $2\sum_{i=1}^{n/2} E[d_i]/n = \Theta(1)$. ■

The following theorem shows that the throughput-delay scaling trade-off provided by Scheme Π is optimal for the static random network model. The proof follows easily from the converse for the fluid model (Theorem 2 in [2]) and hence we do not provide the proof.

Theorem 3: Let the average delay be bounded above by $D(n)$. Then the achievable throughput $T(n)$ for any scheme scales as $O\left(D(n)/n\right)$.

III. Conclusion

The optimal throughput-delay trade-off for static random wireless networks was determined in [1] with a more complete treatment in part I [2] of this work. It was done using a fluid model where the packet size needed to scale down with the number of nodes n in the network. In this paper, we imposed the constraint that the packet size remains constant and established that the throughput-delay trade-off remains unchanged. This also provides a justification for the simplifying fluid assumption made in [1] and [2] since it does not affect the essential network dynamics.

The next natural issue to address would be the mobile random network model, introduced in [3], for which the trade-off was studied using a fluid model in [1]. In the case where the mobile network also has n static nodes along with n mobile nodes, for any throughput $T(n) = \Theta(1/n^{1/2+\epsilon}), \epsilon > 0$, we can show that the trade-off given by $T(n) = \Theta(D(n)/n)$ can be achieved. This is the same as the trade-off for the fluid model in [2]. This establishes the optimal trade-off for this range of low throughput. The scheme achieving this trade-off uses the scheduling scheme given in this paper along with a randomization technique that makes the mobile network appear like a static network. Further in this scheme, packets chase their destinations in a manner similar to Scheme 3(a) in [2].

REFERENCES

[1] A. El Gamal, J. Mammen, B. Prabhakar, D. Shah, "Throughput-Delay Trade-off in Wireless Networks", *IEEE INFOCOM*, Hong Kong, 2004.

[2] A. El Gamal, J. Mammen, B. Prabhakar, D. Shah, "Throughput-Delay Trade-off in Wireless Networks – Part I: The Fluid Model", Submitted to *IEEE Trans. on Information Theory*, preprint available at `www.stanford.edu/ jmammen/papers/TDfluid.ps`.

[3] M. Grossglauser and D. Tse, "Mobility Increases the Capacity of Ad-hoc Wireless Networks", *IEEE INFOCOM*, Anchorage, Alaska, pp.1360-1369, 2001.

[4] P. Gupta and P. R. Kumar, "The Capacity of Wireless Networks", *IEEE Trans. on Information Theory*, 46(2), pp. 388-404, March 2000.

[5] F. P. Kelly, "Reversibility and Stochastic Networks", *Wiley*, 1979.

[6] S. R. Kulkarni and P. Viswanath, "A Deterministic Approach to Throughput Scaling in Wireless Networks", *IEEE Trans. on Information Theory*, 50(6), pp. 1041-1049, 2004.

[7] R. Motwani and P. Raghavan, "Randomized algorithms", *Cambridge Univ. Press*, 1995.

[8] R. W. Wolff, "Stochastic Modeling and the Theory of Queues", *Prentice-Hall*, 1988.