Oblivious Routing with Mobile Fusion Centers over a Sensor Network

Devavrat Shah
Department of Electrical Eng. and Computer Science
Massachusetts Institute of Technology
devavrat@mit.edu

Sanjay Shakkottai
ECE Department
The University of Texas at Austin
shakkott@ece.utexas.edu

***Abstract*— We consider the problem of aggregating data at a mobile fusion center (fusor) (eg. a PDA or a cellular phone) moving within a spatial region over which a wireless sensor network (eg., fixed motes) has been deployed. Each sensor node generates packets destined to the fusor, and our objective is to develop strategies that can route the packets to the mobile fusor.**

For an arbitrary (possibly random) fusor mobility pattern over any connected subset of the sensor deployment area, we first derive upper bounds on the aggregation data rate (i.e., the uniform rate region from each sensor node to the mobile fusor), where we allow all sensor nodes to have complete knowledge of the mobility pattern of the fusor.

We then consider aggregation data rates that can be achieved when the mobility pattern of the fusor is unknown to the sensor nodes. Surprisingly, we show that for a class of mobility patterns (random mobility over connected-compositions of convex sets of the deployment region, e.g. random walks over piece-wise linear sets), we can construct "universal" mobility-oblivious routing strategies that achieve aggregation data rates that are of the same order as the (mobility-aware) upper bound.

I. Introduction

Networks of sensor devices, which have the capabilities of collecting, (wirelessly) forwarding and aggregating (fusing) data are of increasing importance in a wide range of applications. In this paper, we consider the problem of aggregating data at a *mobile fusor* device, such as a handheld PDA which can communicate with nearby sensor nodes located over some geographic region.

Applications involving such mobile fusors over a sensor field are many-fold. For example, a foreman moving around on a civil construction site can use a hand-held device to determine the location/completion status of various objects and construction material (e.g., piping spools or structural steel components) which are sensor-node tagged [1]. Such just-in-time (JIT) management – which asks for small waiting or inspection times – on job sites can help to raise productivity levels [2]. In this scenario, from the networking standpoint, "status" data needs to be routed from sensor nodes to the foreman's handheld. However, the critical problem here is that sensor nodes do not a-priori know the location of the foreman. In a military context, a mobile soldier would need sensor-data from fixed (say, randomly air-dropped) sensor devices over the battlefield. Again, as in the construction example, the location of the soldier may not be necessarily known to the sensor nodes.

A naive approach to addressing the problem of communicating with an "unknown" sink (the mobile fusor) is through flooding. While flooding ensures that the mobile fusor receives the information regardless of its position, the data rate that can be supported is very poor. To see this, consider the following "back-of-the-envelope" calculation.

Example 1 *Consider a wireless sensor network with n nodes randomly placed over a unit area, with each node having a wireless radio of range $r(n)$ (i.e., the node can communicate with all neighbors that are within a distance of $r(n)$).*

Now, it immediately follows that there can be no more than $1/(\pi r^2(n))$ simultaneous transmissions at any time (because each radio transmission "covers" an area of $\pi r^2(n)$). On the other hand, to flood a single packet from a fixed node to all other nodes in the network, there needs to be at-least $1/(\pi r^2(n))$ transmissions (this is because the packet has to "cover" the entire area in order to reach all the nodes). These two observations imply that for all n nodes to support a data rate of $\lambda(n)$ each, we need to satisfy $\lambda(n) \leq \frac{1}{n}$.

Observe that this data rate is extremely low – for instance, with point-to-point routing with known source and destination locations, we know from [3] that $\lambda(n) = \Omega(1/\sqrt{n})$ is achievable, which is a much larger rate.

A. Main Contributions

The above discussion motivates the main questions we address in this paper:

(a) What is the "best" we can do if all sensor nodes possibly *know* the location and mobility pattern of the fusor?
(b) More importantly, can we construct oblivious routing schemes that do not have any knowledge of the location of the fusor, and still manage to achieve data rates that are "close" to the upper bounds? For instance, flooding is an oblivious scheme – however, the data rate with this strategy is very poor.

The results that we present in this paper are listed below.

(i) We first develop an upper bound in this paper for arbitrary mobility patterns. This uses cut-set ideas (i.e., the amount of data "flux" that can enter any set S in the network) to construct the bound.
(ii) For fusors with good mobility patterns (which include mobility along lines, over convex sets, and connected-

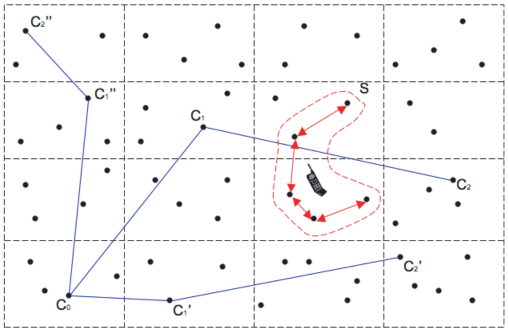

Fig. 1. Two stage oblivious routing. A mobile fusor node moves over a restricted set S to which packets have to be obliviously routed. The paths of different packets are chosen independently. Note that having two stages is critical to ensure that the second hop is uniformly random.

compositions of convex sets, and is formally defined in Section II-B, see also Figures 3 and 4), we construct an oblivious routing strategy that achieves aggregation data rates which are of the same order as the (mobility-aware) upper bound.

The main idea in our routing strategy is for each packet from a sensor node (source) to be "spread" only over some restricted part the network, such that this set "cuts" the mobile fusor's path. This is done by routing along lines to a randomly chosen destination – but along two random stages (see Figure 1). Two stage routing ensures that the *second stage* is "random enough" over the network (note that the first stage path is correlated for all outgoing packets from the sensor source). Our scheme ensures that a "sufficient" number of second-stage lines cross *any* good mobility pattern (and hence, the unknown path of the fusor). Thus, by using appropriate *coding* at the sensor source, we ensure that the fusor can recover the packets from all nodes.

B. Related Work

There has been much interest in the throughput-capacity and delay of large wireless networks. Gupta and Kumar [3] introduced (random) large-scale wireless network models for studying throughput scaling in a static setting. They considered n randomly placed nodes over a region of unit area, each with a wireless communication radius that scaled[1] as $\Theta(\sqrt{\frac{\log(n)}{n}})$, and, each node has an associated randomly chosen destination (i.e., random source-destination pairs). They showed that for large values of n, each source destination pair can support a data rate (through a multi-hop relaying strategy) that scales as $\Theta(1/\sqrt{n \log n})$. Other related work in the context of static networks with different channel models and routing strategies include [4]–[8].

In the context of mobile nodes, [9], [10] have shown that a throughput capacity of $\Theta(1)$ can be supported, by using mobility of nodes intelligently to minimize interference due to packet transmissions, however, at the cost of a large delay.

The results in [3] and [9] have motivated a large body of work that has studied the trade-off between throughput and delay, both in the context of static and mobile networks [11]–[18]. However, to the best of our knowledge, we are not aware of throughput or delay results (either bounds or constructive strategies) that address routing toward a single mobile node that moves only over a restricted region of the domain in a large wireless network.

In the following sections, we provide the needed definitions and summarize our main results. In Section II, we formally describe the models, and define the problem. We then present the main results in Section II-D. Next, in Section III, we present an upper bound on the aggregation data rate at the fusor. Then, we proceed to present an algorithm (Section IV-A) in Section IV, and show that the algorithm is (order) optimal. We finally conclude in Section VI.

II. Setup and Problem Statement

A. Network topology

We are given n sensor nodes randomly placed over an unit area. For concreteness we assume that the unit area is a torus, i.e. a unit square with pair of opposite sides identified. The x-axis of the unit torus constitutes one edge of the square, and the y-axis of the unit torus corresponds to the orthogonal direction (see Figure 3).

Each sensor node is capable of wireless transmission. We assume that a node can communicate with any other node within its transmission range $r(n)$. The following is a well-known property for such network, which follows from an application of Chernoff's bound and Union bound.

Property 1 *Given n nodes thrown uniformly at random on the unit torus, we partition the unit torus by a grid of equal sized square tiles, with each square partition (henceforth referred to as a* cell*) having area $4\log(n)/n$. Then, each cell in the torus has at least one node with high probability*[2]*. Further, each cell contains no more than $10\log(n)$ node w.h.p.*

Given the Property 1, it can be shown that the network is connected w.h.p. if $r(n) = 5\sqrt{\log(n)/n}$. Further each node is not connected to more than $100\log(n)$ nodes w.h.p. The results of this paper will qualitatively remain the same as long as Property 1 (and hence its implication) are satisfied.

However, for ease of explanation, in the rest of this paper, we will restrict our analysis to the random uniform node placement on the torus with $r(n) = 5\sqrt{\log(n)/n}$. Let $G = (V, E)$ denote the graph formed by these n sensor nodes, with V representing vertices and E representing edges.

[1] We use the following notation in this paper: (i) $f(n) = O(g(n))$ means that there exists a constant c and integer N such that $f(n) \leq cg(n)$ for $n > N$. (ii) $f(n) = o(g(n))$ means that $\lim_{n\to\infty} f(n)/g(n) = 0$. (iii) $f(n) = \Omega(g(n))$ means that $g(n) = O(f(n))$, (iv) $f(n) = \omega(g(n))$ means that $g(n) = o(f(n))$. (v) $f(n) = \Theta(g(n))$ means that $f(n) = O(g(n))$; $g(n) = O(f(n))$.

[2] In this paper, with high probability (w.h.p.) refers to a probability at least $1 - 1/n^2$, unless specified otherwise.

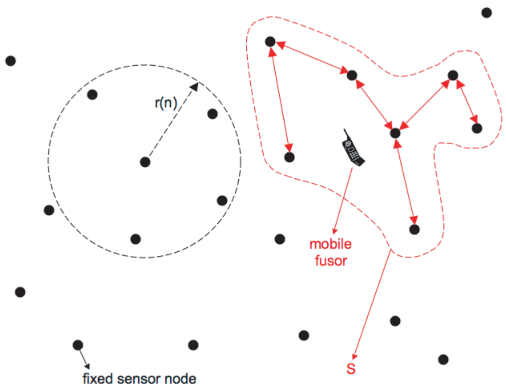

Fig. 2. A mobile fusor node moves over a restricted set S of the fixed sensor nodes V.

B. Model for Traffic, Communication and Mobile Fusor

Traffic Model: All sensor nodes generate new data at rate $\lambda(n)$. The data is generated according to an independent exogenous process, which is assumed to be Poisson (however, it can be any 'friendly' arrival process such as a renewal process with finite second moment).

The data generated by nodes is to be collected by a mobile fusor. Data that is collected or absorbed by the mobile fusor disappears from the network (i.e. departure of data in the standard queueing terminology).

Mobility Model: We consider a discrete-time setup (i.e., time is partitioned into time-slots). At each time-slot, the mobile fusor can communicate with exactly one sensor node. Thus, we denote the position of the mobile fusor by the index of the sensor node it can communicate with. We assume that the mobility pattern of the mobile fusion center (fusor) is described by a random walk (RW) over a subset of V, denoted by S (see Figure 2). We will assume that S is a connected subset (where two nodes are connected with they are within distance $\Theta(\sqrt{\log n/n})$.

The RW is arbitrary over the set S and assumed to have a stationary distribution given by $\boldsymbol{\pi} = (\pi(i))_{i\in S}$ on S. The fusor changes its location along the edges of G according to the RW at discrete time (in this paper, time will be measured in units of the steps of this RW). Let $p(S)$ denote the perimeter[3] of set S.

Communication Model: The mobile fusor can upload data from the sensor node (i.e., extract data) at its current location at a rate $R(n)$. Throughout the paper, we assume that $R(n) = \omega(1)$, that is it scales so that $R(n) \to \infty$. The precise scaling of $R(n)$ will affect $\lambda(n)$ as we shall see later in the paper. The rate required for fusor-to-sensor communication is higher than that of sensor-to-sensor communication, as the fusor has to collect (at each time-slot) all the packets from a large collection of sensor nodes that have aggregated at a sensor node.

Usually, S is a small subset of V (this is the case when it is more realistic and question becomes interesting). Hence sensor nodes need to route their data to S so as to be collected by the fusor. For routing data, nodes need to transmit data to its neighbor so as to spread it via multi-hopping. The transmission is done over a common wireless channel. For successful transmission, we consider the following well-known disk model or Protocol model introduced by Gupta and Kumar [3].

Definition 1 (Protocol model) *A transmission from node i to node j is successful if and only if the following conditions are satisfied: (1) distance between i and j is no more than $r(n)$, and (2) any other simultaneously transmitting node k is at least distance $2r(n)$ away from node j. When transmission is successful, i can transmit unit amount of data to node j.*

In this paper, we will use $r(n) = \Theta(\sqrt{\log n/n})$, which is the radius of connectivity when nodes are thrown at random. An immediate implication of this model is that each node can transmit or receive data from other sensor nodes[4] at most at the unit rate. A useful definition about the capacity of S is given below.

Definition 2 (Cut capacity) *Given a set of nodes $S \subset V$, define the cut-capacity of S as the maximal sum rate at which nodes in S can exchange (transmit or receive) data with nodes in S^c (i.e. nodes not in S). Denote this by $\rho(S)$.*

We state the following result that relates the cut-capacity of S with its perimeter.

Lemma 1 *Consider set S with perimeter $p(S)$. Then, under the Protocol model with $r(n) = \Theta(\sqrt{\log n/n})$, we have*

$$\rho(S) = O\left(p(S)\sqrt{\frac{n}{\log n}}\right).$$

Proof: We need to use the following two facts that are direct implication of Protocol model: (1) two nodes that are farther than $3r(n)$ cannot communicate with each other, (2) among nodes within distance $r(n)$, transmission can happen at rate $O(1)$.

Now to prove the above claim, lay down a square grid Grid-1 of side $2r(n)$. Now, make a copy of this grid, call it Grid-2, shift this grid by $r(n)$ to the left and $r(n)$ downwards. Now, any set with perimeter $p(S)$ cuts at most $O(p(S)/r(n))$ squares of Grid-1 as well as these many squares of Grid-2 for the following simple geometric reason: to cut more than 9 square cells of side $2r(n)$ by any connected curve, the length of the curve must be at least $r(n)$. It is not hard to see that number of nodes that are in S and S^c that can communicate under Protocol model with radius $r(n)$ must belong to the square cells of Grid-1 and Grid-2 which are intersected by perimeter of S. Thus, the total area in which

[3] Precisely, the perimeter of set S is the length of the shortest connected piece-wise linear closed curve (i.e. loop) that is made of line-segments joining vertices of S and geographically enclosing all the points of S.

[4] However, recall that the sensor node can communicate with the mobile fusor at a higher rate of $R(n)$.

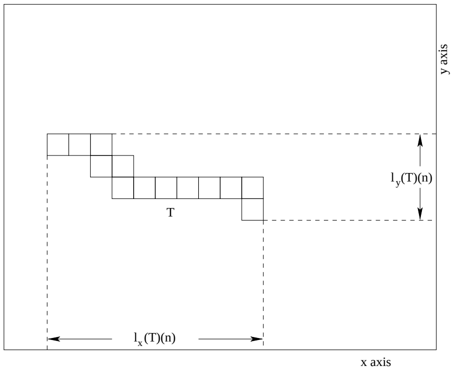

Fig. 3. Example of a good pickup set. The ratio of the perimeter to the larger projection $\max\{\ell_x(\mathcal{T})(n), \ell_y(\mathcal{T})(n)\}$ for a good pickup set is a $\Theta(1)$ quantity.

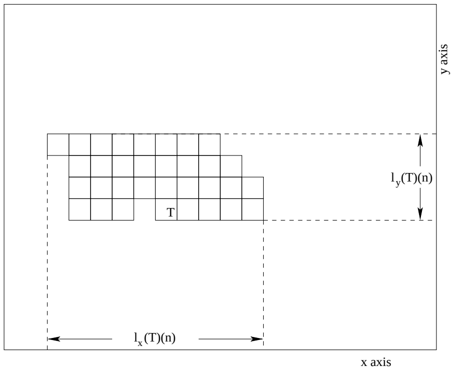

Fig. 4. Example of a good pickup set. The ratio of the perimeter to the larger projection $\max\{\ell_x(\mathcal{T})(n), \ell_y(\mathcal{T})(n)\}$ for a good pickup set is a $\Theta(1)$ quantity.

the nodes that can exchange data between S and S^c belong is at most $O(p(S)r(n))$. By the Protocol model, the data rate per area $\Theta(r^2(n))$ is unit. That is, the total rate at which data transfer can happen between S and S^c is bounded above by $O(p(S)/r(n))$. For $r(n) = \Theta(\sqrt{\log n/n})$ as selected earlier, we have this as $O(p(S)\sqrt{n/\log n})$. This completes the proof of Lemma 1. $\blacksquare$

Finally, we define a notion of equivalence of sets in terms of capacity.

Definition 3 (Good pickup set) *A set S of nodes visited by mobile fusor is called* good *pickup set if there exists a collection of cells, say $\mathcal{T} = \{T_1, T_2, \ldots\}$, such that the following conditions are satisfied:*

(i) **Connectedness condition:** *The cells of $\mathcal{T}$ form a single connected component. Here, we call cells T_i and T_j connected if they are neighbors of each other.*

(ii) **Inclusion condition:** *Sensor node $v \in S$ if $v \in T_i$, for some $T_i \in \mathcal{T}$,*

(iii) **Projection condition:** *Let $\ell_x(\mathcal{T})$ be the length of the projection of $\mathcal{T}$ on the x axis, and $\ell_y(\mathcal{T})$ be the length of the projection of $\mathcal{T}$ on the y axis. Define $\ell(\mathcal{T}) = \max\{\ell_x(\mathcal{T}), \ell_y(\mathcal{T})\}$. Then, $\ell(\mathcal{T}) = \Theta(p(S))$.*

The *inclusion condition* simply states that the set $\mathcal{T}$ consists of cells which constitute the nodes in S.

The *projection condition* states that the perimeter of the set S is of the same order as the longer projection of the set $\mathcal{T}$ on the coordinate axes. For instance, a collection of nodes along a line segment, a “smooth curve,” a “thin” and “long” set or a convex set such as a circle or ellipse are good pickup sets (see Figures 3 and 4). A set such as a comb (see Figure 5), which has a large perimeter compared to its projection is not a good pickup set.

C. Problem Statement

Each sensor node generates new data at rate $\lambda(n)$. The data is generated according to an independent exogenous process, which is assumed to be Poisson (however, can be any ’friendly’ arrival process). The data that is absorbed by mobile fusor disappears from the network (i.e. departure of data in the standard queueing terminology).

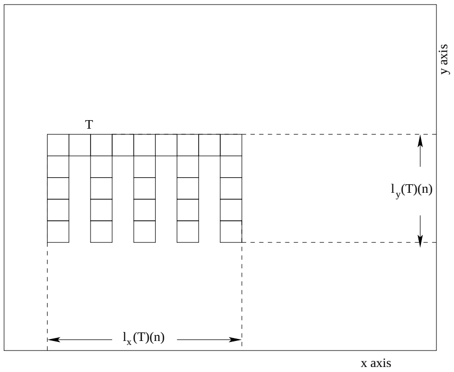

Fig. 5. Example of a bad pickup set. Consider the “top” line of length $\Theta(1)$ (equivalently, $\Theta(\sqrt{n/\log(n)})$ cells). Then, we can have $\sqrt{n/\log(n)}$ downward “teeth” to the comb, each of $\Theta(1)$ length. This implies that the projection on each of the axes is a $\Theta(1)$ quantity. However, the length of the perimeter of the set is $\Theta(\sqrt{n/\log(n)})$. We disallow such shapes in our analysis.

The primary goal in this paper is to identify the maximal supportable rate $\lambda(n)$ (for all sensors) that can be supported by the network infrastructure and the mobile fusor so that net unabsorbed (or undeparted) data in the network remains finite with probability 1. We henceforth refer to this as the *stability* condition.

D. Main Results

The main result of the paper is about characterization of the maximal rate $\lambda(n)$ at which data can be transmitted to S in an *oblivious manner*, i.e. when sensor nodes do not know the location/values of $S, \boldsymbol{\pi}$. Specifically, our interest is in a routing strategy that is oblivious. The precise definition is as follows.

Definition 4 (Oblivious routing) *A routing scheme is said to be oblivious if sensor nodes that lie in the set S^c do not have any knowledge of the location (either current position or statistical distribution) of the fusor.*

The precise results are stated below.

Upper bound. The following is a straight forward upper bound on $\lambda(n)$.

Theorem 2 *For any set S such that $|S| \leq n/2$,*

$$\lambda(n) = O\left(\frac{\rho(S)}{n}\right).$$

To support rate $\lambda(n)$ while keeping system stable, we need

$$R(n) = \Omega(n\lambda(n)).$$

Further, there exists set S supporting the maximal rate $\lambda(n) = \Theta(\rho(S)/n)$, for which it is required that

$$R(n) = \Omega(\rho(S)) = \Omega(|S|/\log n)$$

for rendering the system stable.

We note that the upper bound places no restriction on the fusor location knowledge at sensor nodes. In other words, all sensor nodes can know the location/values of $S, \boldsymbol{\pi}$ of the fusor.

Lower bound. The more surprising result is the following matching (in order) lower bound via a simple oblivious routing scheme. Thus, our scheme is order optimal both in terms of maximal supportable $\lambda(n)$ and $R(n)$ requirement in an *oblivious* manner.

Theorem 3 *There exists a (randomized) oblivious routing and fusor pick-up scheme so that*

$$\lambda(n) = \Omega\left(\frac{\rho(S)}{n}\right),$$

and

$$R(n) = O\left(|S|/\log n\right),$$

if S is a good pickup set with $\boldsymbol{\pi}$ such that $\pi_i = \Theta\left(\frac{1}{|S|}\right)$. More generally, $R(n) = O\left(\frac{1}{\log n(\min_i \pi_i)}\right)$ is sufficient.

III. Upper Bound: Proof of Theorem 2

The proof of upper bound follows using simple arguments: For data generated by nodes outside S, it must enter set S from S^c. Now the number of nodes outside S, i.e. $|S^c|$ is at least $n/2$. Hence, the data rate that needs enter S from outside is $\Theta(n\lambda(n))$ for any supportable $\lambda(n)$.

By definition, the maximal rate at which data can enter S is $\rho(S)$. Hence, the above discussion immediately imply that

$$n\lambda(n) = O(\rho(S)).$$

Equivalently, maximal supportable $\lambda(n) = O(\rho(S)/n)$.

Further, it immediately follows that we require $R(n) = \Omega(n\lambda(n))$. This is because in each time-slot, the sensor nodes generate data (on average) at a total rate of $\Theta(n\lambda(n))$. On the other-hand, the fusor can pick up data at each time-slot at a rate of (at most) $R(n)$.

We next show that there exists set S (with $|S| \leq n/2$) such that for supporting this maximal rate $\lambda(n)$, we require

$$R(n) = \Omega(\rho(S))$$

for rendering the system stable.

Now, consider a set S consisting of a collection of say some M square cells (each cell has side $\sqrt{n/\log(n)}$) on a horizontal line. Then the number of nodes in S is $\Theta(M \log n)$. Then, $\rho(S) = \Theta(M)$. The scheme of Theorem 3 will imply that we can support rate

$$\lambda(n) \;=\; \Theta\left(\frac{\rho(S)}{n}\right) \;=\; \Theta\left(\frac{M}{n}\right).$$

Hence, we require that

$$R(n) = \Omega(n\lambda(n)) = \Omega(\rho(S)) = \Omega(M) = \Omega(|S|/\log n).$$

This completes the proof of Theorem 2.

IV. Lower Bound: Proof of Theorem 3

To establish the proof of Theorem 3, we first present an algorithm which we show will achieve the rate $\lambda(n)$ claimed in Theorem 3.

A. Algorithm

The algorithm has the following parts: *(1)* Routing of data by nodes, *(2)* Scheduling of transmission, and (3) Pick up of data by mobile fusor. These parts are described separately as follows.

Routing. Each node generates data at rate $\mu_r(n) = \Theta\left(1/\sqrt{n \log n}\right)$ according to a Poisson process. Let the node be in cell, say C_0. The node sends packet as follows: first it picks two cells C_1 and C_2 independently and uniformly at random from all possible $\Theta(n/\log(n))$ cells. Then, the packet is first sent from C_0 to C_1 along the straight line joining C_0 and C_1 via hopping along the cells joining them. After reaching cell C_1, route the packet from C_1 to C_2 in the same manner along the straight line[5] (see Figure 1). During routing, if a packet passes through a cell which contains a node $v \in S$, then node v stores a copy of this packet in its FCFS queue. When packet reaches the cell C_2, the packet is destroyed (however its copy is stored in nodes $v \in S$ that were on the packet's path).

Scheduling. A non-trivial task involved along with routing packets in the network is that of scheduling packet transmissions at cells while observing the Protocol model. There are many ways in which this is usually dealt with [3], [13]. We consider the following well-known TDMA protocol. Since transmission of a node from a cell interferes with nodes in $\Theta(1)$ cells, transmission of cells can be scheduled so that each cell gets $\Theta(1)$ fraction of the time to transmit. Further, the time allocated to a cell is uniformly divided among all of its nodes. Thus, each node gets $\Theta(1/\log(n))$ fraction of the time to transmit.

The above description of scheduling is sufficient for establishing the rate feasibility. However, when packets are of fixed size and we wish to have constant delay on average per hop

[5] Strictly speaking, we route to nodes in the network, not cells. However, for ease of notation and discussion, we will say that we route to a cell C_1 if we are route a packet to any node $v \in C_1$.

for packets, a sophisticated scheduling is required for packet transmissions at the nodes. Such a scheme was used in [14]. An analogous scheme can be used in our context as well. We refer reader to Section V and Appendix for further details.

Pick up. The mobile fusor, when present at node $i \in S$, picks up the head-of-line $R(n)$ packets from the FCFS queue at node i. The fusor makes sure that it does not pick up the same packet multiple times. We assume a simple (low-overhead) mechanism in place to ensure this, which can be done in many ways (for example, a handshaking mechanism at the begining of each time-slot where list of packet IDs can be sent by the sensor node to the fusor, based on which packets can be selected for upload to the fusor).

B. Analysis

Our objective is to show that the effective rate achieved by the above scheme is $\Omega\left(\frac{\rho(S)}{n}\right)$. We note that under the above described scheme, the generation rate $\mu_r(n)$ of each node remains the same irrespective of the S. However, the *effective rate* (i.e. number of packets reaching successfully) depends on the properties of S.

Determining this effective rate is sufficient to show that our scheme achieves the desired data rate. This is because sources can do appropriate coding in the following sense. Suppose sources find that only a fraction p of their packets reach the mobile fusor successfully. As we shall soon see, the probability of each packet reaching is identical and independent under our scheme. Hence, the network can be seen as an erasure channel between each source and the mobile fusor with the probability of erasure being $1-p$. Using appropriate codes for an erasure channel such as MDS codes [19] or low-complexity LDPC [20] codes can yield that the source can effectively transmit data at rate p. We refer reader to Section V for some additional related discussion. Hence, in what follows we will be interested in determining effective rate at which packet reach from source to destination.

Effective rate computation. The computation of effective rate is divided into three parts.

(i) First, we establish that the routing scheme is feasible, i.e., the network load induced by the routing scheme can be supported by the $\Theta(1)$ bandwidth that is available at each cell.

(ii) Second, we compute the effective rate at which data enters the set S which satisfies the hypothesis of Theorem 3 under the above described scheme.

(iii) Third, we show that under the hypothesis of Theorem 3, the fusor can be pick up data at sufficient rate to render the system stable.

(i) Feasibility of Routing: Under the routing scheme described above, each node generates data at rate $\Theta(\mu_r(n))$. Thus, each cell generates data at rate $\Theta(\mu_r(n) \log n)$ given Property 1. As described in Section IV-A, each packet is routed in two stages. The first stage consists of the selection of a random cell C_1 uniformly from all $\Theta(n/\log(n))$ cells and routing packet from C_0 to C_1. In a similar fashion, the second stage involves selection of another randomly chosen cell C_2, and routing the packet from cell C_1 to C_2.

Property 2 *Under the two stage routing for each packet, the straight route length joining centers of C_0, C_1 and C_1, C_2 is $\Theta(1)$ with probability at least 0.9. Since each route length is at most $\sqrt{2}$, we have that a packet travels $\Theta(1)$ distance on average during the two-stage routing.*

The Property 2 implies that packet makes $\Theta(\sqrt{n/\log n})$ hops on average during the routing in two stages since each hop makes the packet travel a distance $r(n) = \Theta(\sqrt{\log n/n})$. Now each node generates packets at rate $\mu_r(n) = \Theta(1/\sqrt{n \log n})$. Since each packet makes $\Theta(\sqrt{n/\log n})$ hops on average and there are n nodes, we require that in unit-time the total number of hops (over all flows) made is $\Theta(n\mu_r(n)\sqrt{n/\log n})$, which is $\Theta(n/\log n)$. By Property 1, each cell has $\Theta(\log n)$ nodes. Subsequently, the symmetry of the routing strategy implies that the hops are distributed uniformly (in order). There are total $\Theta(n/\log n)$ cells. Hence, we have a requirement of $\Theta(1)$ hops on average per cell. By selection of an appropriate $\Theta(1)$ scaling constant, this means that we can support the routing algorithm for a rate of $\mu_r(n) = \Theta(1/\sqrt{n \log n})$. This proves the feasibility of routing scheme.

(ii) Effective Rate to S: Given the feasibility of routing, we next compute the effective rate at which data enters the set S. Recall that in our scheme, if a packet traverses through set S, the data is picked up by (one or more) nodes in set S.

Lemma 4 *For a good set S, under the two stage routing, a packet enters a node in S with probability at least $\Omega(p(S))$.*

Proof: From Definition 3 of a good pickup set S, there exists a collection of cells[6] $\mathcal{T}$ such that these cells (and their nodes) are inside S; $\ell(\mathcal{T}) = \Theta(p(S))$. $\ell_x(\mathcal{T}) \leq 1/6$ and $\ell_x(\mathcal{T}) \geq \ell_y(\mathcal{T})$. That is, the whole set $\mathcal{T}$ can be covered in a square of side $1/6$.

Given $\mathcal{T}$, we can now draw a connected curve $\mathcal{C}$ that passes through cells of $\mathcal{T}$ and its projection on x axis, $\ell_x(\mathcal{C}) = \Theta(p(S))$ and $\ell_y(\mathcal{C}) \leq \ell_y(\mathcal{T})$. Let C_s and C_e denote the start point and the end point of $\mathcal{C}$ respectively (see Figure 6). Now, cover the curve within a square box, Bx of side at most $\Theta(p(S))$, which has side smaller than $1/6$ by above discussion. Since torus does not have edge effects, we can shift (translation of origin) the coordinates such that the center of Bx is the same as center of torus.

Now consider two thin horizontal strip St_u, St_d each of height $1/12$ and width of unit length: St_u at vertical distance $1/12$ from the top of Bx above it and St_d at vertical distance $1/12$ from bottom of Bx below it. Now for any point $P \in St_u$, connect it with the end points of $\mathcal{C}$, i.e. C_s and C_e and

[6] In this proof, for ease of discussion (and without loss of generality), we assume that $\ell(\mathcal{T}) \leq 1/6$, and that $\ell_x(\mathcal{T})(n) \geq \ell_y(\mathcal{T})(n)$. For, if $\ell(\mathcal{T})(n) > 1/6$, we can always work with a connected subset of $\mathcal{T}$ which has a projection that is $\leq 1/6$.

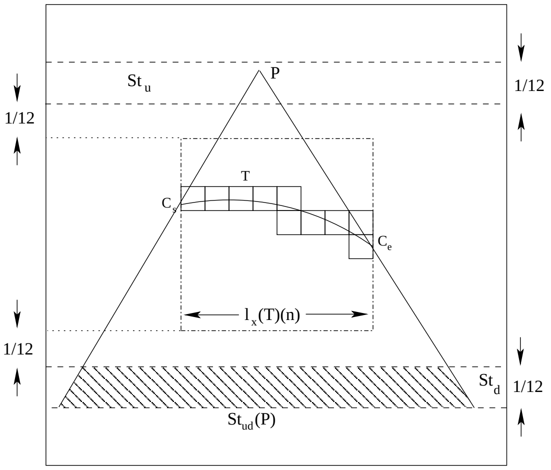

Fig. 6. Illustration of the construction of the region $ST_{ud}(P)$ used in Lemma 4.

extend these lines till they cross the strip St_d completely. Call the region in St_d between these two lines' intersection as $St_{ud}(P)$. From geometry, it is not hard to see that the area of $St_{ud}(P) = \Theta(p(S))$ for any point P. Next, we will use this to determine the probability of packet entering S as $\Theta(p(S))$.

To this end, note that selecting C_1 and C_2 for second stage is same as choosing two points, P, Q uniformly at random in the torus and then selecting the cells C_1 and C_2, where P, Q belong to C_1 and C_2 respectively. Going with this probabilistic interpretation, if we choose P in St_u and Q in $St_{ud}(P)$, then the line joining them will intersect the curve C since under our routing scheme, we are routing along the shortest path joining C_1 and C_2; by construction the shortest path must cross the $\mathcal{C}$. If the line cuts C, then it must cut a cell of $\mathcal{T}$. That is, the packet traveling along that line must enter S. Thus, the probability of entering S is at least the probability that first point P is in St_u and Q is in $St_{ud}(P)$. This is $\Theta(1)$ times $\Theta(p(S))$, which is $\Theta(p(S))$. This completes the proof of Lemma. ■

Given Lemma 4, clearly the probability of each packet reaching S is at least $\Theta(p(S))$. Since each node generates data at rate $\mu_r(n)$, we have effective rate given as

$$\lambda_{\text{eff}} = \Theta(\mu_r(n)p(S)) = \Theta\left(\frac{p(S)}{\sqrt{n\log n}}\right).$$

By Lemma 1, we know that $\rho(S) = O(p(S)\sqrt{\log n/n})$. From this and above equality, we have

$$\lambda_{\text{eff}} = \Omega(\rho(S)/n). \tag{1}$$

Now, by Theorem 2, we know that λ_{eff} can not be larger than $O(\rho(S)/n)$. Thus, we have

$$\lambda_{\text{eff}} = \Theta(\rho(S)/n). \tag{2}$$

(iii) Pickup Rate of Fusor: The above two steps establish that the data can be pushed in the set S at desired effective rate. The remaining task is to show that it can be picked up by the mobile fusor at an appropriate rate from the nodes in S.

To this end, we compute the effective rate at which data is coming in a cell of S. Then we show that mobile fusor, under the hypothesis of Theorem 3 can pick up data at rate higher than this effective rate. This will complete the proof of the desired claim.

Consider a cell that contains a node in S. Under the two stage routing scheme, we now compute the effective rate at which data is passing through this cell. From the computation done to establish feasibility of the routing scheme, it can be seen that the net data comes into a cell at rate $O(1)$. Now, under stationary distribution $\boldsymbol{\pi}$ such that $\pi_i = \Theta(1/|S|)$, the mobile fusor spends $\Theta(\log n/|S|)$ fraction of time in each cell. If we now choose $R(n) = \Omega(|S|/\log n)$, then the fusor will pick up data from each cell at effective rate of

$$R_{\text{eff}} = \Omega\left(\frac{\log n}{|S|} \times \frac{|S|}{\log n}\right) = \Omega(1).$$

In this case, each cell is served at required rate and hence we have a stable system. This completes the proof of Theorem 3.

V. Discussion: Coding & Scheduling

Here we present some more details related to the coding and scheduling schemes that were mention in the description of oblivious scheme described in Section IV.

First, coding. As remarked earlier, coding is required because a packet sent by each source reaches S (and subsequently is picked up by mobile fusor) with probability p independently. Thus, the effective 'channel' between the source and mobile fusor can be seen as an erasure channel with erasure probability $1-p$ (erasures happening independently). Such a channel has been well-studied and many codes with excellent performance are known. Specifically, it has been shown that by using codes like appropriate LDPC codes [20] (and it recent modifications like [21], [22]), with block length of some large N, the source and mobile fusor can communicate at rate $p-\varepsilon_N$, where $\varepsilon_N < \exp(-\alpha N)$ for some positive constant α. The coding and decoding is extremely simple (roughly $\Theta(N\log(N))$ operations). For this reason, by computing the value of p implies the rate as stated in Section IV. We also refer to recent work by [23] on the use of coding in the context of the throughput-delay trade-off for ad hoc networks.

Now, scheduling. This essentially deals with the question of which packet to transmit when a cell gets a transmission-opportunity. The discussion in Section IV establishes that there exists (such as Time-Division-Multiple-Access (TDMA)) scheduling schemes under which the routing scheme is supportable. However, if one wishes to minimize (in order) the average delay experienced by packets, we need more clever schemes. Such schemes were designed in [14]. We describe an adaptation of such a scheme in the Appendix.

VI. Conclusion

In this paper, we have studied the problem of aggregating data at a mobile fusor moving within a restricted geographic region over which a wireless sensor network has been deployed, and where each sensor node generates packets destined to the fusor.

First, we have characterized the maximum data rate that can supported from the sensor nodes to the fusor. For an arbitrary fusor mobility pattern over any connected subset of the sensor deployment area, we have derived an upper bound on the aggregation data rate that can be supported at the fusor, where we allow all sensor nodes to have complete knowledge of the mobility pattern of the fusor.

Next, we have developed an oblivious routing strategy based on multi-hop routing over two random segments. Using this strategy, we have shown that for a class of mobility patterns, we can achieve (order) optimal aggregation data rates.

VII. Acknowledgments

D. Shah was supported by NSF CAREER grant and DARPA ITMANET Program. S. Shakkottai was supported by NSF Grants CNS-0325788, CNS-0347400 and CNS-0519401, and the DARPA ITMANET Program W911NF-07-1-0028.

References

[1] E. Jaselskis, T. Elmisalami, and B. Stephan, “Radio frequency identification tagging: Applications for the construction industry,” report to the Construction Industry Institute, Austin, TX, 2000.

[2] L. S. Pheng and C. J. Chuan, “Just-in-time management of precast concrete components,” *Journal of Construction Engineering and Management*, vol. 127, no. 6, pp. 494–501, 2001.

[3] P. Gupta and P. R. Kumar, “The capacity of wireless networks,” *IEEE Trans. Inform. Theory*, vol. 46, no. 2, pp. 388–404, 2000.

[4] S. R. Kulkarni and P. Viswanath, “A deterministic approach to throughput scaling in wireless networks,” *IEEE Trans. Inform. Theory*, vol. 50, no. 6, pp. 1041–1049, June 2004.

[5] L. Xie and P. R. Kumar, “A network information theory for wireless communication: Scaling laws and optimal operation,” *IEEE Trans. Inform. Theory*, vol. 50, no. 5, pp. 748–767, May 2004.

[6] O. Leveque and E. Telatar, “Information theoretic upper bounds on the capacity of large extended ad hoc wireless networks,” *IEEE Trans. Inform. Theory*, vol. 51, no. 3, pp. 858–865, March 2005.

[7] A. Jovicic, P. Viswanath, and S. R. Kulkarni, “Upper bounds to transport capacity of wireless networks,” *IEEE Trans. Inform. Theory*, vol. 50, no. 11, pp. 2555–2565, November 2004.

[8] S. Subramanian and S. Shakkottai, “Geographic routing with limited information in sensor networks,” in *Proceedings of Information Processing in Sensor Networks*, April 2005, pp. 269 – 276.

[9] M. Grossglauser and D. Tse, “Mobility increases the capacity of ad-hoc wireless networks,” in *Proc. IEEE INFOCOM*, Anchorage, Alaska, 2001, pp. 1360–1369.

[10] S. N. Diggavi, M. Grossglauser, and D. Tse, “Even one-dimensional mobility increases ad hoc wireless capacity,” in *Proc. IEEE ISIT*, Laussane, Switzerland, July 2002.

[11] N. Bansal and Z. Liu, “Capacity, mobility and delay in wireless ad hoc networks,” in *Proc. IEEE INFOCOM*, 2003.

[12] M. J. Neely and E. Modiano, “Capacity and delay tradeoffs for ad-hoc mobile networks,” *IEEE Trans. Inform. Theory*, June 2005.

[13] A. E. Gamal, J. Mammen, B. Prabhakar, and D. Shah, “Throughput-delay trade-off in wireless networks – part I: The fluid model,” *IEEE Trans. on Information Theory, Special Issue on Networking and Information Theory*, 2006.

[14] ——, “Throughput-delay trade-off in wireless networks – part II: Constant packet size,” *To appear in IEEE Trans. on Information Theory*, 2006.

[15] ——, “Optimal throughput-delay trade-off in energy constrained wireless networks,” in *Proc. IEEE ISIT*, Chicago, 2004, p. 439.

[16] G. Sharma, R. Mazumdar, and N. Shroff, “Delay and capacity trade-offs in mobile ad hoc networks: A global perspective,” in *IEEE Infocom*, Barcelona, Spain, April 2006.

[17] X. Lin, G. Sharma, R. Mazumdar, and N. Shroff, “Degenerate delay/capacity tradeoffs in ad hoc networks with brownian mobility,” *IEEE Trans. on Information Theory, Special Issue on Networking and Information Theory*, 2006.

[18] R. Madan and D. Shah, “Capacity-delay scaling in arbitrary wireless networks,” *Allerton Conference on Communication, Control, and Computing*, 2005.

[19] J. V. Lint, *Introduction to Coding Theory*. Springer, 1991.

[20] R. G. Gallager, *Low Density Parity Check Codes*. MIT Press, 1963.

[21] M. G. Luby, M. Mitzenmacher, M. A. Shokrollahi, D. A. Spielman, and V. Stemann, “Efficient erasure codes.” [Online]. Available: citeseer.ist.psu.edu/268151.html

[22] Richardson and Urbanke, “The capacity of low-density parity-check codes under message-passing decoding,” *IEEETIT: IEEE Transactions on Information Theory*, vol. 47, 2001. [Online]. Available: citeseer.ist.psu.edu/richardson98capacity.html

[23] L. Ying, S. Yang, and R. Srikant, “Coding achieves the optimal delay-throughput trade-offs in mobile ad-hoc networks,” preprint, November, 2006.

[24] F. Kelly, *Reversibility and Stochastic Networks*. Wiley, Chichester, 1979.

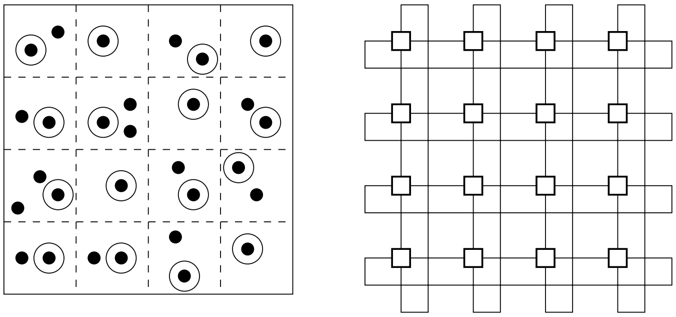

Fig. 7. The torus on the left which has 16 cells and each cell contains at least one node. The circled node in each cell acts as a relay. The corresponding queuing network of 16 cell-nodes (servers), with each server corresponding to a cell in the wireless network, is shown on the right [14].

Appendix
Scheduling Scheme

Now, we describe the schedule scheme that decides which packets to transmit for a cell (this is a direct adaptation of a scheme presented in [14]). To this end, we can view the network of cells as made of $\Theta(n/\log n)$ nodes, one each corresponding to a grid-cell in the original network. Call this network $\mathcal{N}_D$ of $\mathbf{N} = \Theta(n/\log n)$ cell-nodes. An example of such a network is given in Figure 7. Standard arguments for Protocol model imply that each cell-node in $\mathcal{N}_D$ gets to transmit once in $\Theta(1)$ time-slots. For simplicity, we normalize a time-slot so that each node gets to transmit twice in each (normalized) time-slot: one for the “first stage” of routing, which is used only to transmit packets that are being routed in the first stage, and the other for “second” stage of routing which is used only for transmission of packets which are in the second stage of their routing.

In network $\mathcal{N}_D$, each node has its own queue and it is connected to four of its neighbors. Under the routing scheme of Section IV, packets travel from cell nodes corresponding to C_0 to C_1 (resp. C_1 to C_2) via multi-hops through cell nodes on the straight-line joining them in the first stage (resp. second stage). In what follows, we describe scheduling decision for cell nodes for the transmissions corresponding to first stage routing. The scheduling for second stage will follow in the same fashion (with a little difference described at the end).

In the first stage, each node generates data for all other $\mathbf{N}-1$ cells with rate $\Theta(\mu_r(n)/\mathbf{N})$. Also as each cell has precisely $\Theta(\log n)$ nodes, we have that between any two cells, data is generated at rate $\Theta(\log n\mu_r(n)/\mathbf{N})$. Thus, we have effectively $\mathbf{N}^2$ routes, with each having rate $\Theta(\log n\mu_r(n)/\mathbf{N})$. Since the original process is Poisson and the cell C_1 selection is at random, we have that the arrival process for each route is Poisson as well. As argued before, the network is stable, i.e. each cell node can serve all arriving data (with appropriate choice of constants in $\mu_r(n)$). Now, if we had a continuous time network and service discipline being Pre-emptive Last-in-First-Out, denoted as LIFO (PL), then since each packet has deterministic service time (unit normalized time-slot), the queue-size distribution at each node would have been product form[7] with average queue-size being $O(1)$ due to the stability condition of the network. But this network is discrete and hence we need a different policy which is described next. For this, we need some definitions.

Queuing network $\mathcal{N}_C$: Consider a continuous-time open network, with the same topology, routing policy, exogenous arrival process and service requirement for packets, as that of $\mathcal{N}_D$. The only difference is that $\mathcal{N}_C$ operates in continuous time unlike $\mathcal{N}_D$, in which though the arrival process is continuous time, the service happens only at the discrete times. We have Preemptive LIFO (PL) queue management at each server in $\mathcal{N}_C$ (see [24] for more details). As argued above, this network has all the desirable properties: product form distribution, $O(1)$ average queue-size at each node and hence $O(1)$ average delay at each node for a packet and Poisson departure process. Next, will use the simulation of such a network to induce schedule for $\mathcal{N}_D$.

Packet Scheduling in $\mathcal{N}_D$ using $\mathcal{N}_C$: We cannot use the PL policy of $\mathcal{N}_C$ directly in $\mathcal{N}_D$ 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 scheduling policy can be implemented if and only if each packet arrives before its scheduled departure time. According to our scheduling policy, the scheduled departure time can be no earlier than $\lceil a_C\rceil$, whereas the actual arrival time is a_D. Hence for this scheduling policy to be feasible, it is sufficient to show that $a_D \le \lceil a_C\rceil$ for every packet at each server. 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. Since the departure time at a server is the arrival time at the next server on the packet's route, it is sufficient to show 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 ([14]) *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. Also, the remaining packets would have departure times as if they are from 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. ■

Lemma 5 and the already discussed property of $\mathcal{N}_C$ establishes that the average delay per hop for a packet is $O(1)$. Each packet travels at most $O(\sqrt{\mathbf{N}}) = O(\sqrt{n/\log n})$ hops. Thus, the average delay is $O(\sqrt{n/\log n})$.

Finally, scheduling for second stage routing at each node is done in the same way but with the following difference. The packets, after they reach their resp. cell C_1, wait in a special queue till they come out in the $\mathcal{N}_C$ network corresponding to the first stage routing. Then, they are injected back in the network for second stage routing. This will make sure that the arrival process for second stage routing is also Poisson and hence same scheme will work.

[7] This is due to results on product form solutions for quasi-reversible queuing networks. See book by Kelly [24] for details.