Green Wave: Latency and Capacity-Efficient Sleep Scheduling for Wireless Networks

Saikat Guha
Raytheon BBN Technologies,
Cambridge MA, USA
Email: sguha@bbn.com

Chi-Kin Chau
University of Cambridge,
Cambridge, UK
Email: Chi-Kin.Chau@cl.cam.ac.uk

Prithwish Basu
Raytheon BBN Technologies,
Cambridge MA, USA
Email: pbasu@bbn.com

***Abstract*—While scheduling the nodes in a wireless network to sleep periodically can save energy, it also incurs higher latency and lower throughput. We consider the problem of designing optimal sleep schedules in wireless networks, and show the NP-hardness for finding the sleep schedules that can minimize the latency over a given subset of source-destination pairs. We also derive a latency lower bound as $d+O(1/p)$ for any sleep schedules that satisfy the required active rate p (i.e. the fraction of active slots of each node), and d as the shortest path length. Nonetheless, we provide positive results, by a novel solution of optimal sleep scheduling using *green-wave* sleep scheduling (GWSS), inspired by coordinated traffic lights, which is shown to meet our latency lower bound (hence is latency-optimal) for topologies such as the line, grid, ring, torus and tree networks, under low traffic loads. For high traffic loads, we present a non-interfering GWSS, which can achieve the maximum throughput scaling law as $T(n,p) = \Omega(p/\sqrt{n})$ bits/sec on a grid network of size n, with a latency scaling law as $D(n,p) = O(\sqrt{n}) + O(1/p)$. Finally, we extend GWSS to a random network with n Poisson-distributed nodes, for which we show an achievable throughput scaling law as $T(n,p) = \Omega(p/\sqrt{n \log n})$ bits/sec and a corresponding latency scaling law as $D(n,p) = O(\sqrt{n/\log n}) + O(1/p)$; hence meeting the well-known Gupta-Kumar achievable throughput law $\Omega(1/\sqrt{n \log n})$ when $p \to 1$.** [1]

I. Introduction

A primary concern in the design of wireless networks is to ensure acceptably long operational lifetime with the battery-powered wireless nodes. It is well-known that *idle listening* at radio receivers is a major cause of the wasteful energy consumption, since the power consumption in listening mode is comparable to that of *receive* mode. The extant literature has proposed "sleep scheduling" (or duty cycling) of radio transceivers to conserve battery energy, by occasionally turning the transceivers on and off in a controlled fashion [2], [3]. While sleep scheduling attains energy conservation, it is achieved by incurring higher latency and lower throughput. The goal of this work is to investigate the fundamental limits of latency and throughput of sleep scheduling. We aim to address such vital question as: *For a given level of energy consumption rate of nodes in a wireless network, how to schedule the activity of the transceivers to achieve the minimum average end-to-end latency, while meeting certain minimum throughput per source-to-destination (S-D) pair?*

[1]Full version of this paper appears in [1].

Recently, there has been various progress in the design and analysis of sleep scheduling in wireless networks. The latency performance of stateless opportunistic forwarding on finite networks with random and pseudo-random sleep scheduling has been studied in our prior work [4], [5], [6]. These uncoordinated sleep scheduling approaches, while being simple and robust without relying on the centralized coordination, generally suffers from high latency. On the other hand, *coordinated* sleep scheduling approaches can potentially achieve better performance, when provided with the centralized coordination on sleep schedules. Particularly, [7] considers the problem of finding the optimal sleep schedules for minimizing the latency diameter. They showed the problem to be NP-hard (though the proof was later reported as incorrect [8]), and proposed efficient coordinated sleep scheduling schemes for certain specialized topologies.

Despite the progress in prior work, there remain a number of outstanding issues of optimal sleep scheduling in wireless networks. First, how hard is to find the optimal sleep scheduling that, for instance, can minimize the incurred latency? Second, is there a general framework to design sleep scheduling that is optimal in useful common situations? Third, how to ensure these sleep schedules work well in the presence of wireless interference?

This paper addresses the problem of optimal sleep scheduling in a broad and extensive perspective. We provide comprehensive insights on this problem, ranging from the computational complexity, to the limit of latency, and the achievable latency and capacity. Our contributions are outlined as follows.

- (§ II) Considering a slightly harder optimal sleep scheduling problem of [7], but which is more relevant to practical wireless networks, and proving its NP-hardness.
- (§ III) Deriving a tight lower bound on the mean latency for any sleep schedules on arbitrary graphs.
- (§ IV) Proposing a new approach to sleep scheduling inspired by coordinated traffic lights (called green-wave sleep scheduling (GWSS)), which is shown to attain our latency lower bound, and hence latency-optimal under low traffic loads for topologies such as line, grid, ring, torus, and tree networks.
- (§ V) Presenting a non-interfering GWSS for high traffic loads, which can achieve the maximum throughput scaling law as $T(n,p) = \Omega(p/\sqrt{n})$ bits/sec, with a latency

scaling law as $D(n,p) = O(\sqrt{n}) + O(1/p)$, on a grid network with the number of nodes n, active rate p, and $n/2$ S-D pairs.

(§ VI) Extension of non-interfering GWSS to a random network with n Poisson-distributed nodes, for which we show an achievable throughput scaling law as $T(n,p) = \Omega(p/\sqrt{n \log n})$ bits/sec and a corresponding latency scaling law as $D(n,p) = O(\sqrt{n/\log n}) + O(1/p)$; hence meeting the well-known Gupta-Kumar achievable throughput law $\Omega(1/\sqrt{n \log n})$ when $p \to 1$.

II. Delay Efficient Sleep Scheduling

A. Problem Setup

An efficient sleep schedule not only seeks to minimize idle listening, but also to optimize the performance of wireless networks (e.g. latency, capacity) subject to a certain constraint of energy consumption level. This work considers *receiver-based* sleep scheduling (i.e., the *active state* of a node refers to the receiver being active and listening for incoming transmissions), and full topology knowledge at each node.

Let $G \triangleq (V, E)$ be an arbitrary graph. Let $p \in (0, 1]$ be the required active rate. Suppose that time is divided into finite slots, and numbered by $\mathbb{Z}^+ \triangleq \{1, 2, 3, \ldots\}$. Let a scheduling function $f : V \to \mathcal{P}(\mathbb{Z}^+)$ be a slot assignment function that assigns a sequence of active slots[2] to every node. Scheduling function f needs to satisfy the constraint that the fraction of active slots must be equal or smaller than p.

Suppose that node u wants to transmit to an adjacent node v at slot t_0. If $t_0 \in f(v)$, then delay is 0. Otherwise, u has to wait for the earliest slot $t_1 > t_0$, such that $t_1 \in f(v)$, and the delay becomes $t_1 - t_0$. Let $\Delta_{t_0}(u, v)$ be the latency along the minimum-latency path[3] from u to v, given scheduling function f, when the transmission begins at slot t_0. Let us define $\Delta(u, v) \triangleq E[\Delta_{t_0}(u, v)]$ to be the mean latency from u to v under scheduling function f where the expectation is taken over a uniformly distributed $t_0 \in \mathbb{Z}^+$.

Given a subset of source-destination (S-D) pairs $X \subseteq V \times V$, define the latency diameter $D_f(X)$ as the maximum expected end-to-end latency among the S-D pairs in X using the minimum-latency path in G, under scheduling function f. Here, we seek to find the optimal sleep schedules that minimize the latency diameter.

B. Hardness of Delay Efficient Sleep Scheduling

The delay efficient sleep scheduling problem (DESS) was proposed in [7] as an abstraction to study the design of optimal sleep schedules for a given wireless network. Although [7] claimed that DESS is NP-hard, it was reported in [8] that the proof of NP-hardness in [7] contains an error. In summary, [8] made an incorrect reduction from 3SAT problem to DESS, such that a satisfiable instance of DESS does not correspond to a satisfiable instance of 3SAT in the reduction. Hence, the NP-hardness of DESS is still an open question.

In this paper, we study a slightly harder version of DESS, called delay efficient sleep scheduling by selections problem (DESS-S), which is more relevant to practical design of sleep schedules of wireless networks, because it considers a given subset of S-D pairs, instead of any S-D pair as DESS. We show that DESS-S is NP-hard. This presents the evidence of hardness of designing the optimal sleep schedules for arbitrary wireless networks, and motivates the following studies of the optimal sleep schedules in more specific settings.

We consider a specific class of periodic scheduling function $\tilde{f} : V \mapsto \{0, ..., T-1\}$, which assigns a *single* slot for each node to be active in an interval T. We show that even considering only periodic scheduling functions, finding the optimal periodic scheduling function is still hard.

Suppose that each node u starts to transmit at its respective active slot. Given periodic scheduling function $\tilde{f}$, the delay from u to v, for a pair of adjacent nodes $(u, v) \in E$, becomes:

$$\Delta(u,v) \triangleq \begin{cases} T, & \text{if } \tilde{f}(u) = \tilde{f}(v) \\ \tilde{f}(v) - \tilde{f}(u) \mod T, & \text{otherwise} \end{cases}$$

We define the problems of delay efficient sleep scheduling:

1) (*Delay Efficient Sleep Scheduling Optimization Problem (DESS)* [7]) Given a tuple (G, T), find a periodic scheduling function $\tilde{f}$ that minimizes the diameter for all S-D pairs $V \times V$:
$$\tilde{f} = \arg\min_{\tilde{f}'} \left\{ D_{\tilde{f}'}(V \times V) \right\}$$

2) (*Delay Efficient Sleep Scheduling Decision Problem (DESS)* [7]) Given a tuple (G, T, ϵ), decide if there exists a periodic scheduling function $\tilde{f}$ such that for all S-D pairs $V \times V$:
$$D_{\tilde{f}}(V \times V) \leq \epsilon$$

3) (*Delay Efficient Sleep Scheduling by Selections Optimization Problem (DESS-S)*) Given a tuple $(G, T, X \subseteq V \times V)$, find a periodic scheduling function $\tilde{f}$ that minimizes the diameter for the S-D pairs in X:
$$\tilde{f} = \arg\min_{\tilde{f}'} \left\{ D_{f'}(X) \right\}$$

4) (*Delay Efficient Sleep Scheduling by Selections Decision Problem (DESS-S)*) Given a tuple (G, T, X, ϵ), decide if there exists a periodic scheduling function $\tilde{f}$ such that for the S-D pairs in X:
$$D_{\tilde{f}}(X) \leq \epsilon$$

DESS-S is regarded as a slightly harder version of DESS, though DESS-S is more relevant to practical wireless networks, where the S-D pairs are arbitrarily selected, instead of all possible S-D pairs. The proof of the following theorem is given in Appendix.

[2] $\mathcal{P}(S)$ is the *power set* or the set of all subsets of the set S.

[3] The *minimum-latency path* between two nodes in a sleep-scheduled network may not necessarily be identical to, and will in general be only lower bounded by, the *shortest path* between the nodes, i.e., $\Delta_{t_0}(u, v) \geq d_{uv}, \forall u, v \in V$, and $\forall t_0 \in \mathbb{Z}^+$, where d_{uv} is the length (in number of edges) of the shortest path connecting u and v, with equality holding in the absence of sleep schedules ($p = 1$).

Theorem 1: Delay efficient sleep scheduling by selections decision problem (DESS-S) (G, T, X, ϵ) is NP-complete. Hence, delay efficient sleep scheduling by selections optimization problem is NP-hard.

III. Latency Lower Bound

As Sec. II showed that designing optimal sleep schedules on a given network is hard, we next derive a general lower bound on latency for all sleep schedules, which provides us guidance to design optimal sleep schedules for specific cases.

Theorem 2: Let $G \triangleq (V, E)$ be an arbitrary connected graph. Given any scheduling function $f : V \to \mathcal{P}(\mathbb{Z}^+)$, a constant active rate $p \in (0, 1]$ for every node in V, assuming node degree $d_v = O(1), \forall v \in V$ and any S-D pair v and w, the mean latency:

$$\Delta(v, w) \geq d_{vw} + O(1/p), \forall v, w \in V, \tag{1}$$

where d_{vw} is the length of the shortest path connecting v and w. To prove the theorem, we consider the following lemma.

Lemma 1: Given a node $v \in V$, an active rate p, and an arbitrary start time $t \geq 1$, and a scheduling function f, let W_f denote the average number of time slots before v becomes active next. The minimum wait time over all sleep schedules, $\min_f W_f = (1 - p)/2p$.

Proof: (lemma) A sleep schedule f is *T-periodic* if for every node, f assigns sleep and wakeup slots with a periodically recurring pattern with period T. More formally, the sequence $f(v) \mod (T)$ is a periodic sequence. We will first prove the lemma for periodic sleep schedules, and then take the limit $T \to \infty$, so that the proof holds for arbitrary sleep schedules. Let $(v_1, ..., v_T)$ be a T bit binary sequence, with $v_l = 1$ if v is active at l-th time slot under schedule f (i.e., if $l \in f(v)$), and $v_l = 0$ otherwise. By definition $p = \sum_{l=1}^{T} v_l/T$. Let $\vec{x} \triangleq (x_1, ..., x_K)$ be a sequence containing the number of consecutive 0's in between two 1's in $(v_1, ..., v_T)$[4], where $K \geq Tp+1$ is the upper bound on the largest possible number of distinct blocks of consecutive 0's in $(v_1, ..., v_T)$. Hence, the total number of 0's, $\sum_{k=1}^{K} x_k = T(1 - p)$. The average number of time slots to wait before v comes active for a T-periodic sleep schedule f is given by $W_f^{(T)}(\vec{x})$:

$$W_f^{(T)}(\vec{x}) = \frac{1}{T} \sum_{k=1}^{K} \frac{x_k(x_k + 1)}{2}. \tag{2}$$

The minimum wait time is thus given by

$$\min_f W_f = \lim_{T \to \infty} \left[\min_{\vec{x}} \frac{1}{T} \sum_{k=1}^{K} \frac{x_k(x_k + 1)}{2} \right], \tag{3}$$

subject to $\sum_{k=1}^{K} x_i = T(1-p)$. The above can be solved using Lagrange multipliers, and the solution is given by $\min_f W_f = (1 - p)/2p$. The values of x_k that minimize the wait time are all constant, $x_k = (1 - p)T/(Tp + 1)$. Thus, all sleep times must be equitably distributed for the minimum average wait time. ■

Proof: (theorem) Let us say for a given arrival time slot t, the minimum delay path from v to w under f is $P_f(v, w, t)$. If the minimum delay path is identical to the *shortest path* $P_{SP}(v, w)$ for all t (this will be true when p is close to 1), then the lemma readily implies $\Delta(v, w) \geq d_{vw} + (1 - p)/2p \triangleq d_{vw} + O(1/p)$, because the first hop from v to the next node in the path $P_{SP}(v, w)$ will at least take $(1-p)/2p$ slots if we average over a uniformly distributed arrival slot t. If $P_f(v, w, t)$ is not identical to $P_{SP}(v, w)$, it must be longer (in number of edges) that d_{vw} for all t. It is straightforward to show using the arguments above that the minimum value of the mean number of slots before the first nearest neighbor of v comes active, under all choices of schedules f, is given by $(1-p)/(2p\mathrm{d}_v)$ where d_v is the degree of node v. Assuming the node degrees to be $O(1)$ (which is a reasonable assumption for many practical network designs), we again obtain $\Delta(v, w) \geq d_{vw} + O(1/p)$. The following corollary follows readily. ■

Corollary 3: Let $D_f(p)$ be the latency diameter, for a scheduling function f subject to active rate p. Then, $D_f(p) = D(G) + O(1/p)$, where $D(G)$ is the diameter of G.

In the following sections, we will show that an efficient sleep schedule, called green-wave sleep scheduling (GWSS), can attain the end-to-end latency upper bounded by $\Delta(i, j) \leq d_{ij} + O(1/p)$ in certain topologies, and hence prove GWSS to be latency-optimal for those topologies. We remark that the latency lower bound holds for an arbitrary pair of nodes in a sleep-scheduled wireless network, and the difference between the shortest path and the latency of the optimal sleep schedule is only an additive term $O(1/p)$, which does not scale with the size of the network.

IV. Green-Wave Sleep Scheduling (GWSS)

This section introduces the green-wave sleep scheduling (GWSS), which is based on an intuitive concept. If you have ever driven along one of the major avenues of Manhattan at reasonable high speed, you must notice that you never catch a red light. Sequences of green lights move along the avenues at the speed limit, and if you can "ride" on one of those *green waves*, you can drive along with little delay. *Green waves* have been studied extensively in the context of optimizing traffic lights [9], and telephone routing in a square grid [10]. Our sleep schedule is inspired by green wave traffic-light scheduling. Each intersection on the one-way Manhattan avenue is analogous to a sleep-scheduled node, where green and red lights correspond to the active and asleep states.

A *green wave* is defined to be a moving sequence of consecutive active states, moving along a connected path in a graph. A green wave moves at the rate of one hop (i.e., the next consecutive node along the path) in each successive time slot. A (g, r) *periodic green wave* is a periodic sequences of

[4]Note that some v_k's will be zero, when there are consecutive 1's. The total number of 1's is Tp.

g-node active states followed by r-node asleep states moving along a path one hop per time slot. We define g to be the "wake length" and r to be the "sleep length" of the (g, r) periodic green wave. Thus the periodicity of such a periodic green wave is $T = g + r$. See Fig. 1 for an illustration.

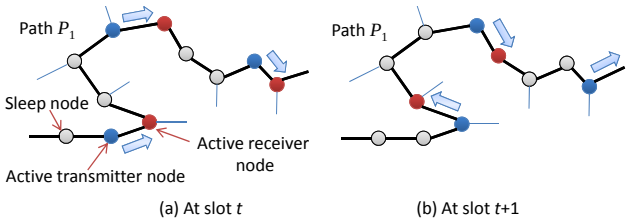

Fig. 1. An illustration of $(1, 3)$ green wave moving along the path P_1.

The notion of green wave provides a general framework to construct efficient sleep schedules. Given a graph G, we pick a subset of paths in G, such that every node lies on at least one path. Then for any path, we can schedule two green waves: (1) a (g, r) left-moving periodic green wave (LGW), and (2) a (g, r) right-moving periodic green waves (RGW) that travels in the reverse direction as LGW on the path[5]. Hence, a packet can ride on either green wave depending on the direction, and can be forwarded with minimal latency. This forms the *green-wave sleep scheduling* (GWSS) schemes. Although [7] proposes similar specific schemes for specialized topologies, GWSS is a general framework that can be applied to arbitrary network topologies.

Particularly, we describe GWSS for the following topologies, and present the respective latency analysis under low traffic load (without the contentions for the same receiver), which is shown to meet our latency lower bound.

1) (*Line Network*): There is one path; we can schedule LGW and RGW travelling in opposite directions. From the analysis in the technical report [11], we can show that the latency for a pair nodes i, j is

$$\begin{aligned}\Delta(i,j) &= d_{ij} + \frac{(1-p)+\sqrt{1-p}}{2p} \\ &= d_{ij} + O(1/p) \end{aligned} \quad (4)$$

And the expected latency averaged over all node pairs is given by

$$\begin{aligned}E[\Delta(i,j)] &= \frac{n+1}{3} + \frac{(1-p)+\sqrt{1-p}}{2p} \\ &= \frac{n+1}{3} + O(1/p) \end{aligned} \quad (5)$$

2) (*Grid Network*): We label a 2D grid network with $m \times m$ nodes from $(1, 1)$ to (m, m) where $n = m \times m$ where n is the total number of nodes. For each horizontal path between $(t, 1)$ and (t, m), we schedule LGW and RGW travelling in opposite directions. For each vertical path between $(1, s)$ and (m, s), we schedule an up-moving periodic green wave (UGW) and a down-moving periodic green wave (DGW) travelling in opposite directions. See Fig. 2 for an illustration.

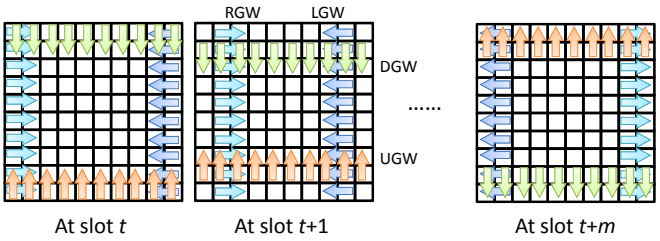

Fig. 2. An illustration of green waves moving on a grid.

From the analysis in the technical report [11], we can show that the latency for a pair nodes i, j is

$$\begin{aligned}\Delta(i,j) &= \frac{r(2r+1)}{6(r+1)} + \frac{r}{2} + d_{ij} \\ &= d_{ij} + \frac{r(5r+4)}{6(r+1)}. \end{aligned} \quad (6)$$

where

$$p = \frac{(2r^2+2r+1)(2r+1)}{(1+r)^4} \quad (7)$$

Hence, this implies

$$\Delta(i,j) = d_{ij} + O(1/p), \quad (8)$$

And the expected latency averaged over all node pairs is given by

$$\begin{aligned}E[\Delta(i,j)] &= \frac{2(m+1)}{3} + \frac{r(5r+4)}{6(r+1)} \\ &= \frac{2(m+1)}{3} + O(1/p). \end{aligned} \quad (9)$$

3) (*Tree Network*): Pick a root in a tree network called c_0. There is only one path from the root to any leaf node. Hence, we can schedule LGW and RGW travelling in opposite directions for each path between the root and a leaf node. The latency analysis of tree network follows similarly from line network, and can be shown to be meeting our latency lower bound. See the full technical report for the detailed calculation [11].

The cases for ring and torus networks follow similarly from the ones of line and grid networks.

By the latency lower bound in Theorem 2, GWSS is latency-optimal on these network topologies. Therefore, for an arbitrary pair of nodes in these sleep-scheduled networks, the difference between the shortest path and the latency using the GWSS scheme is just an additive $O(1/p)$, and does not scale with the size of the network.

[5]The notions of left and right are mainly for distinguishing the two green waves, which may not reflect the physical orientation.

V. Latency and Capacity Scaling Laws for Grid

We have shown that on several lightly-loaded specialized network topologies such as line, grid, ring, torus and tree networks, by careful scheduling of green waves, the mean latency for any S-D pair (i, j) averaged over all routing start times t_0 is given by $\Delta(i, j) = d_{ij} + O(1/p)$. In this section, we consider the more realistic case with high traffic loads, in which the presence of wireless interference will affect both latency and throughput of sleep-scheduled wireless networks. We show that, despite wireless interference, GWSS can be carefully designed to support the well-known Gupta-Kumar capacity scaling law, without incurring extra latency.

A. Interference Models

Let us consider a square grid network with $n = m \times m$ equal-sized square cells on a square plane with one node located at the center of each square cell (see Fig. 3). We denote the length of one side of the square cell as c. Let us randomly choose a set X of $n/2$ S-D pairs whose coordinates are given by (t_i, r_i), $i \in X$. We denote $\mathcal{S} \subseteq X$ as the set of simultaneous links. There are two common interference models to formulate $\mathcal{S}$ in the literature [12]:

1) *Pairwise (or protocol) interference model*: for $i, j \in \mathcal{S}$,

$$|t_j - r_i| \geq (1 + \delta)|t_i - r_i|$$

2) *Aggregate (or physical) interference model*: for all $i \in \mathcal{S}$,

$$\frac{\mathsf{P_{tx}}|t_i - r_i|^{-\alpha}}{\mathsf{N_0} + \sum\limits_{j \in \mathcal{S} \setminus \{i\}} \mathsf{P_{tx}}|t_j - r_i|^{-\alpha}} \geq \beta$$

Lemma 2: It is well-known [12], [13] that if each transmission is limited to communication between adjacent cells on the grid network, then there exists a constant κ (independent of n), such that simultaneous transmissions can take place among links that are κ cells away, without violating both pairwise and aggregate interference models, and can achieve a throughput of $\Omega(1)$ on each simultaneous transmission. The value of κ only depends on the parameters in the interference model (e.g. $\delta, N_0, \mathsf{P_{tx}}, \alpha, \beta$).

We define the per-node throughput $T(n, p)$ as the number of bits per second that with high probability (w.h.p.) all random $n/2$ S-D pairs can communicate at simultaneously. The latency $D(n, p)$ is the number of hops needed for a given packet starting at a source node to reach the destination node, averaged over all S-D pairs. If full topology information is available and in the absence of sleep schedules, the optimal route between any S-D pair is one of the (potentially several) "L"-shaped minimum-distance *Manhattan routes*. But in the presence of sleep schedules, a minimum-distance Manhattan route may or may not be the minimum-latency path, depending upon the underlying sleep scheduling scheme and the active rate p. Below we consider two different sleep scheduling schemes and calculate their capacity and latency scaling laws.

B. Receiver-based Round-Robin Sleep Scheduling

The (sender-based) round-robin scheduling has been commonly used in [12], [13] for showing the achievable capacity law. Here, we modify such scheduling to be receiver-based, which can be compatible with receiver-based sleep scheduling.

Assume that each S-D pair routes the packets on the respective minimum-distance Manhattan routes. We group a block of k^2 cells into a macro-cell as shown in Fig. 3 with $k > \kappa$, and use round-robin scheduling on all $\Theta(n)$ macro-cells simultaneously, such that simultaneously transmitting cells are k cells away. Each receiver node in a macro-cell will be active in every k^2 slots (while all other cells in the macro-cell remain asleep in that slot), and accepts transmissions from four neighboring cells as shown in Fig. 3 sequentially in four *sub-slots*. So, each receiver's active slots has a periodicity of k^2 slots, and the overall active rate $p = 1/k^2$. The conditions for Lemma 2 apply here because in each sub-slot, the simultaneously transmitting cells will transmit to their respective 1-hop neighbors and they will be *tiled* the exact same way as the receiving cells are.

The maximum latency from one macro-cell to the next macro-cell is k^2 slots. It can be shown that the number of macro-cells in the end-to-end S-D path is $\Theta(\sqrt{n})$ w.h.p. when $n \to \infty$ [12]. Hence the end-to-end latency of a packet scales as $D(n, p) = \mathrm{O}(\sqrt{n}/p)$. Notice that, the poor latency performance of this scheme is due to the fact that a packet has to wait for $\mathrm{O}(1/p)$ slots on an average at *each* macro cell.[6] At each slot, there are $\Omega(np)$ transmissions, and the hop count between a S-D pair is $\Theta(\sqrt{n})$ w.h.p. when $n \to \infty$. Hence, the throughput per S-D pair is $T(n, p) = \Omega(\frac{np}{n \cdot \sqrt{n}}) = \Omega(p/\sqrt{n})$. Therefore, we have:

Theorem 4: Using receiver-based round-robin sleep scheduling, the end-to-end latency of a packet scales as $D(n, p) = \mathrm{O}(\sqrt{n}/p)$, while the throughput per source-destination scales as $T(n, p) = \Omega(p/\sqrt{n})$.

C. Non-Interfering Green-Wave Sleep Scheduling

To overcome the latency problem of round-robin sleep scheduling, we propose non-interfering green-wave sleep scheduling (NiGWSS) – different than the simple GWSS in Sec. IV. Each individual S-D pair still routes data on the shortest-path Manhattan route. Consider a right-moving *tiled green wave front* as shown in Fig. 4(a), such that the receivers of the nodes in blue cells remain active simultaneously and the active states move forward to the orange cells in the next slot, and so on. Notice that each horizontal line of the grid sees a periodic green wave with active states occurring after every k^2 slots. As is evident from Fig. 4(a), for $k > \kappa$, the conditions of Lemma 2 hold because each simultaneous transmission effected by the green wave is one cell to the right. Now let us divide each slot into four sub-slots as shown in Fig. 4(c), and

[6] In fact, with light load, a latency scaling of $\mathrm{O}(\sqrt{n}/p)$ can be obtained by a simple pseudo-random duty cycling approach as described in [4] with opportunistic forwarding along the shortest path.

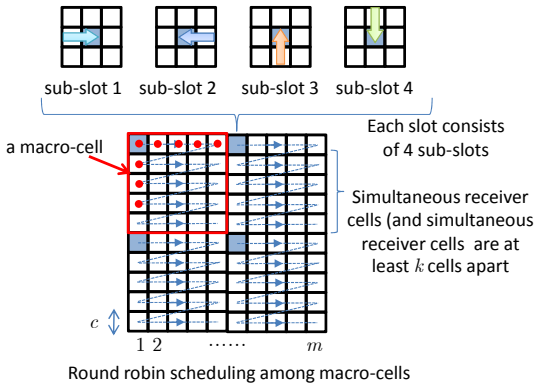

Fig. 3. An illustration of receiver-based round-robin sleep scheduling with $k = 5$, and $n = 100$. The red dots at the center of each square cell are the nodes.

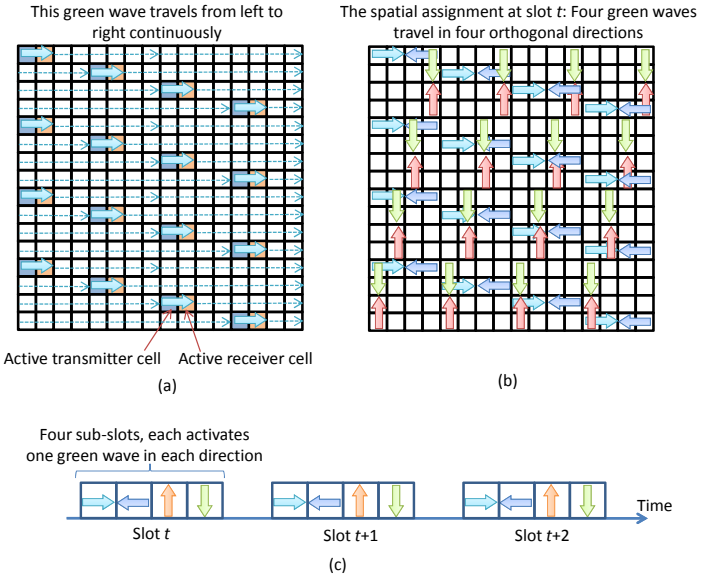

Fig. 4. An illustration of non-interfering green-wave sleep scheduling (NiGWSS) on square grid (with $k = 4$) with optimized latency and throughput performance.

consider four tiled green-wave fronts traversing the network continuously from the four orthogonal directions, as shown in Fig. 4(b). Each successive sub-slot activates one out of the four green waves. Hence in each sub-slot the no-interference condition of Lemma 2, $k > \kappa$, prevails.

The latency of a packet along an end-to-end S-D path, whose length is $\Theta(\sqrt{n})$ w.h.p. when $n \to \infty$, is given by four components: (i) The average number of time-slots a packet has to wait at the source node in order to *hop on to* the green wave moving along the horizontal arm of the Manhattan route, (ii) the length of the horizontal arm of the Manhattan route, (iii) waiting at the corner of the Manhattan route for the next green wave moving along the vertical arm of the Manhattan route to arrive, and finally (iv) the length of the vertical arm of the Manhattan route. Therefore $D(n,p) = \mathrm{O}(1/p) + \Theta(\sqrt{n}) + \mathrm{O}(1/p) + \Theta(\sqrt{n})$, or $D(n,p) = \mathrm{O}(\sqrt{n}) + \mathrm{O}(1/p)$. This proves that the lower bound to the optimal latency scaling in the presence of sleep schedules (by Theorem 2) is achievable. Hence, NiGWSS achieves the optimal latency performance for the square grid network – even in the presence of multiple simultaneous flows.

Let us concentrate on a single S-D pair, and the corresponding transmission of data along a Manhattan route under NiGWSS. The source chunks up the data into small-sized packets that can be reliably transmitted to a one-hop neighbor. The source sends out each packet sequentially on a green wave, which reaches its final destination in $\mathrm{O}(\sqrt{n}) + \mathrm{O}(1/p)$ slots. But in between transmitting each successive packet, the transmitter has to wait k^2 slots. Hence the effective per S-D pair throughput attained is given by $T(n,p) = \Omega(p/\sqrt{n})$ bits/s. Therefore, we obtain the following theorem.

Theorem 5: Using non-interfering green-wave sleep scheduling (NiGWSS), the end-to-end latency of a packet scales as $D(n,p) = \mathrm{O}(\sqrt{n}) + \mathrm{O}(1/p)$, while the throughput per S-D pair scales as $T(n,p) = \Omega(p/\sqrt{n})$.

Note that we choose k given the requirement on the active rate p, such that $k^2 \approx 1/p$. One obvious variation to the above scheme is to choose the separation between active nodes in the horizontal and vertical directions to be different, say k_1 and k_2 respectively. This results in $p = 1/(k_1 k_2)$. In order to realize an arbitrary active rate over time, one can switch between using two sets of green-wave sets described by the tuples $\{k_1, k_2\}$ and $\{k_1', k_2'\}$, such that each of the factors k_1, k_2, k_1' and k_2' are chosen be be larger than the critical radius κ. The interference model does not permit using the aforementioned GWSS with an active rate $p > 1/\kappa^2$ because of excessive interference. But as $p \to 1$, we approach the Gupta-Kumar scaling laws [12], $T(n) = \Omega(1/\sqrt{n})$, and $D(n) = \mathrm{O}(\sqrt{n})$, which can be achieved by the TDMA scheme recently proposed by Franceschetti, et. al. [13].

VI. Latency and Capacity Scaling Laws for Random Network

A random *extended* network is obtained by placing nodes with a unit-intensity Poisson point process over the square region $B_n \triangleq [0, \sqrt{n}] \times [0, \sqrt{n}]$. The questions we ask in this section are; in a random extended network, for a sub-unity active rate $p < 1$, (i) What are the ultimate limits and tradeoffs between throughput and latency scaling laws? and, (ii) are there achievable sleep scheduling schemes that can achieve those ultimate limits of throughput and latency for a sleep-scheduled network? Even though we do not answer the above questions completely in this paper, we examine a few constructive sleep scheduling schemes for random extended networks and the corresponding scaling-law operating points.

A. The Gupta-Kumar Regime

Consider the $\sqrt{n} \times \sqrt{n}$ square area B_n with n nodes and divide it up into a grid with square cells of side-length $\sqrt{\log n}$.

Then, there will be at least one node in each cell w.h.p., which can act as the representative relay node for the rest of the nodes in that cell, and following the constructive proof of Gupta and Kumar [12], the per hop capacity can be shown to be $T(n) = \Omega(1/\sqrt{n \log n})$, as $n \to \infty$. The per S-D pair latency (in hops) scales as the number of cells along one side of B_n, i.e. $D(n) = \mathrm{O}(\sqrt{n/\log n})$. Using the non-interfering green-wave sleep scheduling (NiGWSS) developed in Sec. V, it is straightforward to extend this to a constructive sleep scheduling scheme for a sub-unity active rate p. The per node throughput and per S-D pair latency are given by $T(n,p) = \Omega(p/\sqrt{n \log n})$, and $D(n) = \mathrm{O}(\sqrt{n/\log n}) + \mathrm{O}(1/p)$ respectively.

B. The FDTT Regime

Consider again the $\sqrt{n} \times \sqrt{n}$ square area B_n with n nodes. Recently, Franceschetti et. al. [13] closed an important gap in the capacity of wireless networks [12], by showing that for random extended networks of n randomly located nodes, a $T(n) = \Omega(1/\sqrt{n})$ bit rate per S-D pair can always be achieved, with an average number of hops $D(n) = O(\sqrt{n})$. The FDTT scheme is a constructive protocol that divides up B_n into a square grid with cells of a constant side-length c, such that the probability of a given cell not containing any node is less than e^{-c^2}. In [13], a percolation theory based argument is used to embed in the network an approximate multi-hop grid topology composed of a set of almost equally spaced out, wavy but non-intersecting horizontal and vertical *percolation highways*, followed by pairwise coding-decoding at each hop on these highways coupled with an embedded TDMA slotted transmission schedule to suppress interference from simultaneous transmissions.

There are two possible ways to extend the FDTT scheme to incorporate sleep scheduling, or node active rates $p < 1$:

1) Transmitter-based Round Robin Sleep Scheduling: The figure on the right in Fig. 5 shows the cells that transmit on the same time slot. This TDMA transmitter-based round robin approach controls interference, and is key to deriving the throughput scaling laws in [13]. What the authors do not explicit note is that with this round-robin schedule, each receiver node does not need to stay active in every time slot. With the underlying percolation highway grid, each node can expect to receive data only from a maximum of four possible neighboring nodes on an intersecting pair of vertical and horizontal highways. So, with a $k = 4$ schedule (as shown in Fig. 5), a mean node active rate of $p = 4/k^2 = 0.25$ is embedded in the scheme. It is straightforward to use the techniques we developed in Sec. V, to generalize the FDTT routing scheme for the random extended network to lower values of p, by increasing the side-length k of the transmit macro-cells, but holding $d = 1$ hop transmissions on the highways in each time slot to be constant. It is straightforward to show that throughput and latency scaling laws given by $T(n,p) = \Omega(p/\sqrt{n})$ and $D(n,p) = \mathrm{O}(\sqrt{n}/p)$ are readily attained. As noted in Sec. V-B, under light load conditions, this latency scaling can be trivially obtained with no topology knowledge just by using opportunistic forwarding along the shortest paths.

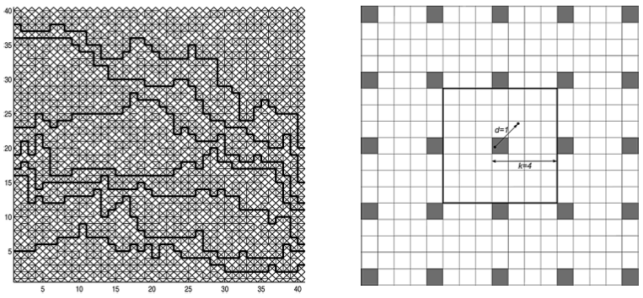

Fig. 5. This figure from [13] on the left shows computer-generated horizontal *highways* or connecting paths in a 40×40 bond percolation grid. The small square cells are of side-length c, and cells that are open are ones with no nodes ($P(\text{no node}) < e^{-c^2}$). The cells that are marked with a horizontal or a vertical line have at least one node in them that act as a relay node on the highway. The figure on the right shows the same cells (of side c – now drawn 45 degrees rotated) with the gray-shaded ones allowed to transmit on the same time slot.

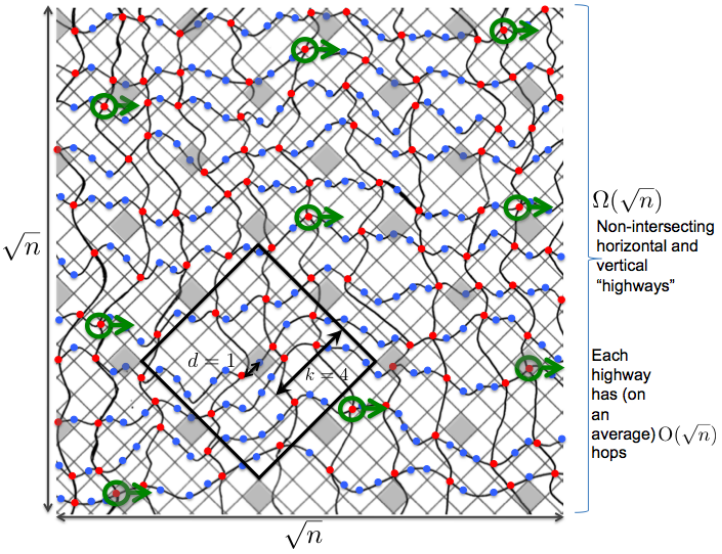

Fig. 6. Horizontal and vertical percolation "highways" criss-crossing the network area B_n containing n Poisson-distributed nodes.

2) Green Wave Sleep Scheduling: Now the natural question to ask is, whether GWSS or a variation thereof (such as NiGWSS) can lead to better latency performance on a random extended network, like it does for the grid network as we found in Sec.V-C. Let us take a closer look at the percolation highways criss-crossing the square region B_n, as shown in Fig. 6. There are $\Omega(\sqrt{n})$ non-intersecting highways crossing the network area in each direction (i.e., two horizontal or two vertical highways never pass through the same cell). Also, the average number of hops on each highway scales as $\mathrm{O}(\sqrt{n})$ as $n \to \infty$. Moreover, the highways can be divided into disjoint sets of $\lceil \epsilon \log \sqrt{n} \rceil$ paths, contained in a $\sqrt{n} \times b \log \sqrt{n}$ rectangular slice of B_n, for all $b > 0$ and ϵ sufficiently small. In Fig. 6, The red nodes are where a horizontal and

a vertical highway intersect. The blue and the red nodes are "backbone" relay nodes along the highways. Notice that each cell that a highway crosses has exactly one backbone node and no two highways cross the same cell. Consider a 'tiled' right-moving green wave front as described earlier in Fig. 4, but hinged only on the red backbone nodes. Also consider four similar tiled green waves moving in four directions, traversing the network with no inter-green-wave interference due to TDMA-transmission in four sub-slots of each time slot (see Fig. 4). The latency scaling achieved using this scheme can be $D(n,p) = \mathrm{O}(\sqrt{n}) + \mathrm{O}(1/p)$. The capacity scaling law analysis runs into interference problems because of a lack of (to our knowledge) a lower bound on the Euclidean distance between two adjacent percolation highways in this framework. Using a variant of this non-interfering GWSS scheme, similar to the square-grid network, it is likely that a per node capacity and latency scalings of $T(n,p) = \Omega(p/\sqrt{n})$ and $D(n,p) = \mathrm{O}(\sqrt{n}) + \mathrm{O}(1/p)$ respectively can be obtained for the random extended network, though a loose lower bound to throughput scaling, $T(n,p) = \Omega(p/n)$ can be shown easily.

VII. Related Work

This paper provided several analytical insights on the limitation and achievable settings of energy-saving sleep scheduling in wireless networks. We note that there are multitude of approaches of energy management in wireless networks, including topology management and network layer optimization [14], [15]. Particularly relevant to our work are those based on duty cycled MAC protocols, which aim to reduce redundant radio operations in MAC protocols, such as 1) idle listening (keeping radio on even when no reception), 2) overhearing (reception of a message not intended for the receiver), and 3) protocol overhead (redundant headers or signalling messages). Examples include S-MAC, SEEDEX, O-MAC, RI-MAC, DW-MAC [2], [16], [17], [18], [19]. Also, our prior work [4], [5], [6] study the approach of reducing idle listening and overhearing by duty cycles based on pseudo-random sequence [3]. As compared to these studies, this paper mainly presents the analytical fundamental results of optimal sleep scheduling problem considering both latency and capacity.

VIII. Discussion and Conclusion

Sleep scheduling wireless transceivers has been established to be a crucial means to improve upon their operational lifetimes, and hence the usefulness of wireless networks. By carefully designing the sleep schedules, this paper shows how the latency and capacity of sleep-scheduled wireless networks can be achieved up to certain scaling laws. Although the general problem of designing delay efficient sleep schedules is NP-hard, we propose to use green-wave sleep scheduling (GWSS) as a general framework for sleep scheduling in wireless networks, which we show to be latency-efficient and capacity-efficient on grid networks, and to be able to maintain good balance between latency and capacity over Poisson-distributed random extended networks.

Especially, we have shown that on large networks with various specialized topologies, GWSS affords almost the same end-to-end latency performance as non-sleep-scheduled networks for moderate values of p, and that the latency is primarily governed by the shortest path distance from source to destination. Also, if only a small number of packets are being sent around a large network (say with a square grid topology), the same $d+O(1/p)$ latency can be more or less maintained for each packet, without increasing p. So, under low traffic loads, increasing the load initially will not result in an increase in energy consumption. Moreover in the low load scenario, if the network is large (i.e. the shortest path lengths d for all S-D pairs are large), then sleep scheduling can save a substantial of energy with only an additional $O(1/p) \ll d$ delay for each transmission. In Sec. V we show that in highly loaded dense networks, throughput decays linearly with p. Hence, as we go from low load to high load, the throughput scaling (with p) transitions (multiplicatively) from $O(1)$ to $O(p)$. In future work, we will investigate the various regimes in this transition space.

Acknowledgments

This research was sponsored by the U.S. Army Research Laboratory and the U.K. Ministry of Defence and was accomplished under Agreement Number W911NF-06-3-0001. The views and conclusions contained in this document are those of the author(s) and should not be interpreted as representing the official policies, either expressed or implied, of the U.S. Army Research Laboratory, the U.S. Government, the U.K. Ministry of Defence or the U.K. Government. The U.S. and U.K. Governments are authorized to reproduce and distribute reprints for Government purposes notwithstanding any copyright notation hereon.

References

[1] S. Guha, P. Basu, C.-K. Chau, and R. Gibbens, "Green wave sleep scheduling: Optimizing latency and throughput in duty cycling wireless networks," *to appear in IEEE J. on Selected Areas in Communications*, 2011.

[2] W. Ye, J. Heidemann, and D. Estrin, "Medium access control with coordinated adaptive sleeping for wireless sensor networks," *IEEE/ACM Trans. Networking*, vol. 12, pp. 493–506, June 2004.

[3] J. Redi, S. Kolek, K. Manning, C. Partridge, R. Rosales-Hain, R. Ramanathan, and I. Castineyra, "Javelen: An ultra-low energy ad hoc wireless network," *Ad Hoc Networks Journal*, vol. 5, no. 8, 2008.

[4] P. Basu and C.-K. Chau, "Opportunistic forwarding in wireless networks with duty cycling," in *Proc. of ACM Workshop on Challenged Networks (CHANTS)*, September 2008.

[5] C.-K. Chau and P. Basu, "Exact analysis of latency of stateless opportunistic forwarding," in *Proc. IEEE INFOCOM*, April 2009.

[6] P. Basu and S. Guha, "Effect of topology knowledge on opportunistic forwarding," BBN Techologies, Tech. Rep., 2009.

[7] G. Lu, N. Sadagopan, B. Krishnamachari, and A. Goel, "Delay efficient sleep scheduling in wireless sensor networks," in *Proc. IEEE INFOCOM*, April 2005.

[8] M. J. Miller, "An errata for delay efficient sleep scheduling in wireless sensor networks," University of Illinois at Urbana-Champaign, Tech. Rep., 2005.

[9] E. Brockfeld, R. Barlovic, A. Schadschneider, and M. Schreckenberg, "Optimizing traffic lights in a cellular automaton model for city traffic," *Physical Review E*, vol. 64, 2001.

[10] G. Kortsarz and D. Peleg, "Traffic-light scheduling on the grid," *Discrete Applied Mathematics*, vol. 53, pp. 211–234, 1994.

[11] S. Guha, P. Basu, and C.-K. Chau, "Green wave: Latency and capacity-efficient sleep scheduling for wireless networks," Computer Laboratory, University of Cambridge, Tech. Rep., 2009.
[12] P. Gupta and P. R. Kumar, "The capacity of wireless networks," *IEEE Trans. Information Theory*, vol. 46, no. 2, pp. 388–404, 2000.
[13] M. Franceschetti, O. Dousse, D. N. C. Tse, , and P. Thiran, "Closing the gap in the capacity of wireless networks via percolation theory," *IEEE Trans. Information Theory*, vol. 53, pp. 1009–1018, March 2007.
[14] J. Pan, Y. T. Hou, L. Cai, Y. Shi, and S. X. Shen, "Topology control for wireless sensor networks," in *Proc. ACM MOBICOM*, 2003.
[15] M. Rossi, L. Badia, and M. Zorzi, "Energy and connectivity performance of routing groups in multi-radio multi-hop networks," *Wireless Communications and Mobile Computing*, vol. 8, pp. 327–342, March 2008.
[16] R. Rozovsky and P. R. Kumar, "SEEDEX: A MAC protocol for ad hoc networks," in *Proc. ACM MobiHOC*, 2001.
[17] H. Cao, K. Parker, and A. Arora, "O-MAC: A receiver centric power management protocol," in *Proc. IEEE ICNP*, 2006.
[18] Y. Sun, O. Gurewitz, and D. Johnson, "RI-MAC: a receiver-initiated asynchronous duty cycle MAC protocol for dynamic traffic loads in wireless sensor networks," in *Proc. ACM SenSys*, 2008.
[19] Y. Sun, S. Du, O. Gurewitz, and D. Johnson, "DW-MAC: a low latency, energy efficient demand-wakeup MAC protocol for wireless sensor networks," in *Proc. ACM MobiHOC*, 2008.

IX. Appendix

Proof: (*Theorem 1*) It is easy to show that DESS-S is in NP, by running polynomial-time Dijkstra's algorithm to determine the minimum-latency path and end-to-end latency of all nodes in G for a given periodic scheduling function f.

To show DESS-S is NP-hard, we rely on a polynomial time reduction from 3SAT problem.

Consider a 3-CNF formula F consisting m clauses and h variables, i.e. $F = c_1 \wedge c_2 \wedge \cdots \wedge c_m$, where each $c_i = y_{j_1} \vee y_{j_2} \vee y_{j_3}$ and $y_{j_1}, y_{j_2}, y_{j_3} \in \{x_1, \bar{x}_1, ..., x_h, \bar{x}_h\}$. F is said to be satisfiable, if there exists a truth assignment to F, such that every clause has at least one true literal. 3SAT is well-known to be NP-complete. Given a 3-CNF formula F, we assume each clause does not contain a literal and its complement (as this is trivially satisfiable).

We next construct a corresponding DESS-S (G, T, X, ϵ), such that F is satisfiable, if and only if (G, T, X, ϵ) is satisfiable. First, we set $T = 2$. So, for $(u, v) \in E$, if $\tilde{f}(u) = \tilde{f}(v)$, then $\Delta(u, v) = 2$. Otherwise, if $\tilde{f}(u) \neq \tilde{f}(v)$, then $\Delta(u, v) = 1$.

We construct G as follows:

1) For each variable $x_j, \bar{x}_j$, we add two nodes $\mathsf{x}_{j,1}, \mathsf{x}_{j,2} \in V$, and an edge $(\mathsf{x}_{j,1}, \mathsf{x}_{j,2}) \in E$.
2) For each clause c_i, we add two nodes $\mathsf{c}_{i,1}, \mathsf{c}_{i,2} \in V$, and add two edges for each variable in c_i:
   a) $(\mathsf{x}_{j,1}, \mathsf{c}_{i,1}), (\mathsf{x}_{j,1}, \mathsf{c}_{i,1}) \in E$, if x_j is present in c_i.
   b) $(\mathsf{x}_{j,2}, \mathsf{c}_{i,1}), (\mathsf{x}_{j,1}, \mathsf{c}_{i,2}) \in E$, if $\bar{x}_j$ is present in c_i.
3) For each pair of clauses c_i and c_{i+1}, we add three nodes $\mathsf{z}_{i,1}, \mathsf{z}_{i,2}, \mathsf{z}_{i,3} \in V$, and four edges $(\mathsf{c}_{i,1}, \mathsf{z}_{i,3}), (\mathsf{c}_{i,2}, \mathsf{z}_{i,3}), (\mathsf{z}_{i,3}, \mathsf{z}_{i,2}), (\mathsf{z}_{i,2}, \mathsf{z}_{i,1}) \in E$.

See Fig. 7 for an illustration of G for a given F.

We next construct the subset of S-D pairs X as:

1) For each clause c_i, we add $(\mathsf{c}_{i,1}, \mathsf{c}_{i,2}) \in X$.
2) For each pair of clauses c_i and c_{i+1}, we add $(\mathsf{z}_{i,1}, \mathsf{c}_{i,1}), (\mathsf{z}_{i,1}, \mathsf{c}_{i+1,1}) \in X$.

It is easy to see that the construction of G and X is of polynomial time, given F in 3-CNF formula representation.

There are two immediate observations:

1) For all S-D pairs in X, the shortest paths in G take at least 3 hops.
2) For a 3-hop path (u_0, u_1, u_2, u_3), we write the minimum end-to-end latency as $\Delta(u_0, u_3)$. $\Delta(u_0, u_3) = 3$ can only be achieved by:
   a) $\tilde{f}(u_0) = 0, \tilde{f}(u_1) = 1, \tilde{f}(u_2) = 0, \tilde{f}(u_3) = 1$, or
   b) $\tilde{f}(u_0) = 1, \tilde{f}(u_1) = 0, \tilde{f}(u_2) = 1, \tilde{f}(u_3) = 0$

   Therefore, we set $\epsilon = 3$.

(*If-Part*): We show that if F is satisfiable, then DESS-S $(G, T = 2, X, \epsilon = 3)$ is satisfiable. First, we set $\tilde{f}(\mathsf{c}_{i,1}) = 0$ and $\tilde{f}(\mathsf{c}_{i,2}) = 1$, $\tilde{f}(\mathsf{z}_{i,1}) = 1$, $\tilde{f}(\mathsf{z}_{i,2}) = 0$ and $\tilde{f}(\mathsf{z}_{i,3}) = 1$ for i. Then, $\Delta(\mathsf{z}_{i,1}, \mathsf{c}_{i,1}) = \Delta(\mathsf{z}_{i,1}, \mathsf{c}_{i+1,1}) = 3$.

Next, if x_j is true, then we set $\tilde{f}(\mathsf{x}_{j,1}) = 0$ and $\tilde{f}(\mathsf{x}_{j,2}) = 1$. Otherwise, we set $\tilde{f}(\mathsf{x}_{j,1}) = 1$ and $\tilde{f}(\mathsf{x}_{j,2}) = 0$. From the construction of G, we know that if c_i is satisfiable, then there exists at least one setting of $\tilde{f}(\mathsf{x}_{j,1})$ and $\tilde{f}(\mathsf{x}_{j,2})$, such that $\Delta(\mathsf{c}_{i,1}, \mathsf{c}_{i,2}) = 3$, in the 3-hop path $(\mathsf{c}_{i,1}, \mathsf{x}_{j,1}, \mathsf{x}_{j,2}, \mathsf{c}_{i,2})$ when x_j is true, or $(\mathsf{c}_{i,1}, \mathsf{x}_{j,2}, \mathsf{x}_{j,1}, \mathsf{c}_{i,2})$ when $\bar{x}_j$ is true.

(*Only-If-Part*): We show that if DESS-S $(G, T = 2, X, \epsilon = 3)$ is satisfiable, then F is satisfiable. First, we know that $\Delta(\mathsf{z}_{i,1}, \mathsf{c}_{i,1}) = \Delta(\mathsf{z}_{i,1}, \mathsf{c}_{i+1,1}) = 3$. This implies that $\tilde{f}(\mathsf{c}_{i,1}) \neq \tilde{f}(\mathsf{z}_{i,1})$ and $\tilde{f}(\mathsf{c}_{i,1}) \neq \tilde{f}(\mathsf{z}_{i+1,1})$. Since $T = 2$, we have $\tilde{f}(\mathsf{z}_{i,1}) = \tilde{f}(\mathsf{z}_{i+1,1})$ for all i. Without loss of generality, we assume $\tilde{f}(\mathsf{c}_{i,1}) = 0$ for all i.

Then, we also know that $\Delta(\mathsf{c}_{i,1}, \mathsf{c}_{i,2}) = 3$. From the construction of G, this is only possible when there exists a 3-hop path:

(1) $(\mathsf{c}_{i,1}, \mathsf{x}_{j,1}, \mathsf{x}_{j,2}, \mathsf{c}_{i,2})$, such that for some j, and $\tilde{f}(\mathsf{c}_{i,2}) = 1, \tilde{f}(\mathsf{x}_{i,2}) = 0, \tilde{f}(\mathsf{x}_{j,1}) = 1$; or
(2) $(\mathsf{c}_{i,1}, \mathsf{x}_{j,2}, \mathsf{x}_{j,1}, \mathsf{c}_{i,2})$, such that for some j, and $\tilde{f}(\mathsf{c}_{i,2}) = 1, \tilde{f}(\mathsf{x}_{i,1}) = 0, \tilde{f}(\mathsf{x}_{j,2}) = 1$.

Since the two cases are exclusive with each other, there is a consistent assignment of x_j, if we set x_j as true for Case (1), and x_j as false for Case (2), which can satisfy each clause c_i.

Therefore, we show that DESS-S is NP-hard, because 3SAT problem is NP-complete. ∎

X. Appendix II

In this appendix, we analyze the latency of GWSS on the line topology. Latency analyses of other topologies can be found in full technical report [11].

A. Latency of Line Network

Consider a (g, r) left-moving periodic green wave (LGW) and a (g, r) right-moving periodic green wave (RGW) on a line network. Assuming that a packet knows the network topology, it knows knows which green waves (LGW or RGW) to choose. In either case, the mean number of time slots T_w the packet needs to wait before it can ride on a green wave is given by

$$\begin{aligned} E[T_w] &= \frac{1}{(g+r)} \sum_{l=1}^{r} l = \frac{r(r+1)}{2(g+r)} \\ &= \frac{r}{2}, \quad \text{for } g = 1. \end{aligned} \tag{10}$$

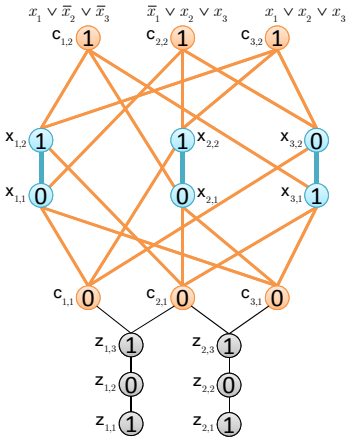

Fig. 7. An illustration of construction of DESS-S $(G, T = 2, X, \epsilon)$ for a given $F = (x_1 \vee \bar{x}_2 \vee \bar{x}_3) \wedge (\bar{x}_1 \vee x_2 \vee x_3) \vee (x_1 \vee x_2 \vee x_3)$. The truth assignment is $x_1 = 0, x_2 = 0, x_3 = 1$. The value in each node indicates the value of the corresponding periodic scheduling function $\tilde{f}$.

Once the packet rides one green wave, in the absence of any other traffic, it needs $d_{ij} = |j - i|$ additional time slots to reach the destination node j, without having to wait at any node in the path. Here d_{ij} is the length of the shortest path from i to j. Hence, the total end-to-end latency is given by

$$\Delta(i,j) \quad = \quad E[T_w] + d_{ij} = d_{ij} + \frac{r}{2}. \tag{11}$$

Let us calculate the mean active rate p for each node. Define $\eta \triangleq g/(g+r)$ as the fraction of time slots that a node is kept active by one (g, r) periodic green wave. A receiver node may be kept active either (i) by the LGW alone, (ii) the RGW alone, or (iii) by both LGW and RGW when they are *overlapping* at that node. Assuming the LGW and the RGW to be statistically independent, the probability that a node is active in a given time slot p is given by the sum of the probabilities of events (i) and (ii) minus the probability of event (iii), i.e.,

$$\begin{aligned} p \quad &= \quad \eta + \eta - \eta^2 = \frac{g(g+2r)}{(g+r)^2} \\ &= \quad \frac{2r+1}{(r+1)^2}, \quad \text{for } g = 1. \end{aligned} \tag{12}$$

Hence for $g = 1$, $r = ((1-p)+\sqrt{1-p})/p$. Hence, the latency $\Delta(i,j)$ is given by

$$\begin{aligned} \Delta(i,j) \quad &= \quad d_{ij} + \frac{(1-p)+\sqrt{1-p}}{2p} \\ &= \quad d_{ij} + O(1/p), \end{aligned} \tag{13}$$

which is identical to the delay bound obtained by Lu et. al. [7] for tree and grid topologies using complicated multi sleep scheduling schemes. Therefore, the expected latency over all node pairs [6] is given by

$$\begin{aligned} E[\Delta(i,j)] \quad &= \quad \frac{n+1}{3} + \frac{(1-p)+\sqrt{1-p}}{2p} \\ &= \quad \frac{n+1}{3} + O(1/p). \end{aligned} \tag{14}$$

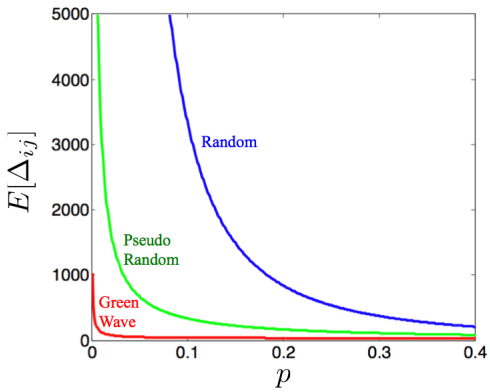

Fig. 8. Latency in line network with n = 100 nodes as a function of active rate p, averaged over a uniform selection of source-destination node pair.

Fig. 8 shows that the coordinated green wave scheme (Eq. (14)) outperforms random and pseudo-random duty cycling schemes at low values of p.