Probabilistic Power Management for Wireless Ad Hoc Networks

Zongpeng Li, Baochun Li
Department of Electrical and Computer Engineering
University of Toronto
{*arcane, bli*}*@eecg.toronto.edu*

***Abstract*— Extending system lifetime by effectively managing power on participating nodes is critical in wireless ad hoc networks. Recent work has shown that, by appropriately powering off nodes, energy may be significantly saved up to a factor of two, especially when node density is high. Such approaches rely on the selection of a *virtual backbone* (i.e., a connected dominating set) of the topology to forward ongoing traffic, coupled with algorithms to manually and periodically recompute such a backbone for load balancing purposes. The common drawback of such schemes is the need to involve periodic message exchanges and to make additional restrictive assumptions. This paper presents *Odds*[1], an integrated set of energy-efficient and fully distributed algorithms for power management in wireless ad hoc networks. Odds build on the observation that explicit and periodic re-computation of the backbone topology is costly with respect to its additional bandwidth overhead, especially when nodes are densely populated or highly mobile. Building on a fully probabilistic approach, Odds seek to make a minimum overhead, perfectly balanced, and fully localized decision on each node with respect to when and how long it needs to enter standby mode to conserve energy. Such a decision does not rely on periodic message broadcasts in the local neighborhood, so that Odds are scalable as node density increases. Detailed mathematical analysis, discussions and simulation results have shown that Odds are indeed able to achieve our objectives while operating in a wide range of density and traffic loads.**

I. Introduction

Wireless ad hoc networks consist of untethered nodes that communicate with each other over multiple wireless hops, with participating nodes collaboratively forwarding ongoing traffic. Since the participating nodes are untethered and usually mobile, energy conservation is critical to extending the lifetime of a functioning network. It has been recognized in recent work [1], [2], [3], [4], [5] that, by entering standby (or deep sleep) mode when the node is idle, significant energy may be saved (usually up to a factor of two), especially when node density is high.

[1]Odds — a shorter term for "probability".

To explore this possibility, the principle concept in the proposed approaches is to establish a *virtual backbone* of the ad hoc network with a small subset of participating nodes. Such a virtual backbone is usually a connected dominating set [6] of the network, and nodes participating in the backbone act as representatives when it comes to forwarding ongoing traffic. All nodes away from the backbone may enter standby mode to conserve energy, if they do not participate as sources or destinations of active connections. The effectiveness of this concept builds on the observation that nodes consume significant power not only when they are sending or receiving packets, but also when they are idle or overhearing ongoing traffic. It has been identified that energy dissipation of an idle node may reach over half of an actively transmitting node with maximum range of power. Further, when the node density (redundancy) is high in the network, it has been observed that each backbone node has the capacity of forwarding ongoing traffic as a representative of the local neighborhood within its range. Connectivity is still maintained due to the natural routing redundancy when the node density is sufficiently high. These observations provide sufficient incentives to design algorithms to turn off all locally and spatially redundant nodes in order to conserve energy.

Existing work [2], [3] following this concept uses periodic message broadcasts in the local neighborhood to drive election algorithms to make local decisions on if a node should participate in the virtual backbone. In GAF (Geographical Adaptive Fidelity) [2], geographic information of each node is assumed to be available via location sources such as GPS, and geographic states of neighboring nodes are periodically exchanged to construct the backbone. In Span [3], periodic HELLO messages are locally broadcast, and are used to drive local algorithms for the nodes to elect or remove themselves from the backbone.

The periodic exchange of local broadcast messages has provided the convenience of designing algorithms based on full knowledge of the states of neighboring nodes. The backbone constructed by such localized algorithms is able to guarantee connectivity. However, we note that such periodic messages illustrates, in essence, a *trade-off* between bandwidth overhead and backbone computation (which leads to energy conservation of non-backbone nodes). We believe that, any schemes involving periodic and local broadcasts of messages *do not scale well* as node density increases. Since each node needs to broadcast a message during each broadcasting interval, as the number of nodes increases, the number of such broadcast messages will eventually increase to the level that may saturate the residual capacity of the network. This may lead to collisions and disruptions to ongoing data traffic. Such collisions may prevent periodic messages to be successfully exchanged, thus hindering the functional correctness and performance of backbone election algorithms. We further emphasize that the benefits of this category of energy conserving protocols — by turning radios off for non-backbone nodes — can only be realized when the node density is sufficiently high, and such benefits should escalate linearly with a linear increase of node density. However, the exacerbating effects of saturating the channel capacity prevent such benefits to scale well. Because of the correlation between scalability of the design and its benefits, this problem is critical and should be considered when designing such algorithms. In short, in addition to the restrictive assumptions (such as the availability of geographic information, and relaying nodes may not be destinations), we believe that periodic exchanges of local states are costly with respect to bandwidth, and are not scalable when node density increases.

Recent work [4] proposes to wake up nodes that are necessary for traffic forwarding only, and let other nodes go to sleeping mode. While such a scheme reduces the periodic message exchange overhead, it pays a price in terms of new session setup latency, since nodes along the transmission route of the new session are very likely to be in the sleeping mode.

Addressing these issues, this paper presents Odds, an integrated set of fully distributed algorithms for power management and energy conservation in ad hoc networks. Odds adopts the same principle that nodes conserve the most energy when they may enter standby mode whenever possible. The design of Odds recognizes the main benefits of electing a virtual backbone of power-on nodes — that of forwarding traffic in an aggregated fashion when there exists sufficient node density and routing redundancy. Building on these design principles, we make several key observations based on rigorous mathematical analysis and discussions. First, it is not necessary to provide hard guarantees with respect to the full connectivity of the entire virtual backbone, as long as (1) all nodes forwarding active ongoing traffic are on the backbone; and (2) the rest of the backbone covers all nodes in a single hop *with a high probability*. Second, in order to compute a backbone that is connected with a high probability, no per-node state information needs to be explicitly and periodically exchanged within the local neighborhood. Third, the topology and traffic in wireless ad hoc networks — especially considering node mobility — are *inherently probabilistic*, there exist few algorithms that are able to guarantee topological or traffic properties such as link and bandwidth availability. This proves it natural (and without leading to inferior performances) to design algorithms following a probabilistic approach. Finally, we assert that the design of any energy conservation algorithms needs to be *flexible* and *adaptive*, so that trade-offs among different parameters are fully considered in different situations — from very light traffic to much heavier loads, from a sparse network to high node densities. Particularly, we need to consider the trade-off between the setup latency of a new connection and the quality of the backbone: with a fully connected backbone the latency is reduced, but it leads to more participating nodes and less energy conservation.

The design of Odds seeks to realize the potential benefits from the above observations by an integrated set of randomized algorithms, building upon provable properties involving *probability*. The main contributions of Odds are the following. First, since they avoid periodic message broadcasts, Odds guarantee scalability to very high node densities, which guarantees the benefits of energy conservation in dense ad hoc networks. The lack of periodic broadcasts also helps the performance of Odds when the nodes are mobile. Second, Odds guarantee that active ongoing traffic is forwarded with similar performance as an always-on network. Third, Odds are flexible: by adjusting the parameters, Odds can present the desired performance in a wide range of network situations with respect to network sizes, node densities, traffic loads and node mobility. Finally, Odds are compatible with standards: they are implemented as extensions to the IEEE 802.11 MAC layer standard, and work well with unmodified on-demand ad hoc routing protocols such as Dynamic Source Routing (DSR) [7] or Ad-hoc On-demand Distance Vector Routing (AODV) [8]. This promotes relatively straightforward deployment in ad hoc networks.

The remainder of the paper is organized as follows. Sec. II reviews the IEEE 802.11 Power Save Mode, that our work is based upon. Sec. III presents Odds with detailed discussions and mathematical analysis. Sec. IV presents extensive results to evaluate the performance of

Odds in the ns-2 network simulator. We conclude the paper in Sec. V.

II. Power Save Mode in IEEE 802.11

Towards the direction of powering off nodes periodically, the wireless LAN MAC layer specifications in the current IEEE 802.11 standard support a *Power Save Mode* (PSM) [9] that conserves energy on idle nodes, by powering their wireless interface off for selected periods of time. Such a Power Save mode is applicable to both infrastructure networks (Basic Service Set), and ad hoc networks (Independent Basic Service Set). In the specification (hereafter referred to as 802.11 PSM), each station may be in one of two power modes: *awake* (when the node is fully powered) and *doze* (when the node is not able to transmit or receive, and consumes very little power). The local clocks of nodes that are in the Power Save mode are synchronized by periodic *beacon* messages. Each beacon message marks the beginning of a *beacon interval*. At this time, a node should suspend all its backoff timers, and after a random backoff period, attempt to send a beacon message if it has not yet received such beacons from other nodes in the local neighborhood. The one who first transmits the beacon message will prevent others to send any further beacons during the corresponding beacon interval.

Each beacon interval starts with an Ad Hoc Traffic Indication Message (ATIM) window. Any data packets destined to a dozing node will be buffered at the upstream neighbor until the upcoming ATIM window. During the ATIM window, all nodes in the local neighborhood will have already been clock-synchronized, and will be awake during the same time interval. Buffered packets will be advertised during the ATIM window by special ATIM frames that we refer to as *advertisements*. Advertisements for unicast data packets need to be acknowledged, while those for broadcast packets do not. A node that has acknowledged previous advertisements should stay awake for one beacon interval in order to receive buffered data packets from the sender. Successfully advertised (and acknowledged, for unicast packets) data packets can be transmitted after the ATIM window. Packets not successfully delivered will be retried during the next beacon interval.

Using such a pure 802.11 PSM approach, all participating nodes in the ad hoc network stay in the *Power Save* mode. Consider the setup latency of a new route from the point of view of an on-demand ad hoc routing protocol. Since both unicast and broadcast packets need to be buffered and subsequently advertised before actual transmissions, the setup latency for a new route will be inevitably high. It may be trivially derived that such setup latency is on the magnitude of kT, where k is the number of hops on the route, and T is the length of a beacon interval. One may believe that once the connection is established, all nodes on the route may stay awake during the lifetime of the connection. This is not true without modifications to the current IEEE 802.11 PSM specification. In the current protocol, packets buffered at an upstream node are only *marked* when an acknowledgment from the corresponding downstream node is received in the ATIM window, and only marked packets can be transmitted in the same beacon interval after the ATIM window. Therefore, a packet that arrives between two ATIM windows has to wait until the next ATIM window to be advertised and marked. This way, the average end-to-end latency of all data packets is on the same magnitude as the setup latency of a new connection. Since $k \propto O(\sqrt{n})$, where n is the number of nodes in the network, it is not scalable to the network size. Odds seek to improve both the average end-to-end latency and the connection setup latency.

III. Odds: Algorithms and Analysis

Odds are an integrated set of distributed algorithms that, if executed in ad hoc network nodes, seek to conserve energy by turning off the radio as frequently and as long as possible. Odds inherit and seek to surpass the current state-of-the-art, by building upon the design principles and key observations illustrated previously. Similar to Span, Odds uses IEEE 802.11 PSM as its base; but different from Span, Odds position itself as an extension to 802.11 PSM, and do not extend itself to the other layers[2]. Finally, the outcome of all algorithms in Odds are probabilistic, favoring minimum protocol overhead when making a trade-off between achievable properties and incurred overhead.

We first discuss properties of an idealized power management algorithm, hereafter referred to as *Utopia*. Utopia is able to predict the arrival of a data packet or MAC frame. On forwarding (or sending/receiving) a packet, it powers itself on exactly when a packet arrives, and powers itself off exactly when it departs the node. If we define the term *normalized energy consumption* as the amount of energy used for each byte of data being forwarded, it is obvious that Utopia can achieve the lowest normalized energy consumption than any realistic algorithms.

As an ideal benchmark for later discussions, we note that Utopia presents the following properties. First, in

[2]In theory, Span may also support routing protocols without modifications. In practice, Span implementation is coupled with a geographic forwarding protocol that needs location information of nodes.

a network with no traffic, Utopia does not consume any power on any of the nodes, assuming the power dissipation level of standby mode is zero. Second, the connection setup latencies of any new connections are identical to that of an always-on network. Third, the end-to-end latency of each data packet is the same as an always-on network. This is critical for any energy conservation schemes, since if energy conservation affects performance (such as latency), it will be unlikely to be utilized. In the first property, the parameter of interests is *energy consumption*; in the second, it is the *connection setup latency*; while it is the *end-to-end latency* in the third. In some cases the benefits of using Utopia are different — and sometimes conflicting — from each other. It is the goal of Odds to be *flexible* and *adaptive*, so it may be able to approach the perfect performances of Utopia in a wide range of different situations with respect to network traffic load and node density.

A. Odds: Overview and Preliminaries

Odds are a collection of algorithms that position themselves as extensions to the standard IEEE 802.11 MAC layer with Power Save mode (PSM). It is a "drop-in" replacement of 802.11 MAC. Compared with IEEE 802.11, it maintains an identical interface to other layers such as routing protocols. For this reason, Odds work with any on-demand routing protocol without the needs of modifications.

All algorithms in Odds use a probabilistic approach to minimize any additional overhead. Such a design is based on the argument that in an ad hoc network with inherently random network behavior, a probabilistic approach is best suitable for algorithms that are *best effort* in nature. We believe that energy conservation algorithms are exactly such *best effort* algorithms in that we need to conserve energy *as much as possible* with minimum disruption on network performance.

The basic assumption of Odds is that each node i runs in the basic IEEE 802.11 Power Save mode. After Odds algorithms are started, the basic 802.11 PSM is used for a short period of stabilizing time. This guarantees the basic properties of 802.11 PSM, that include: (1) synchronized clocks and beacon intervals; and (2) synchronized ATIM windows.

Odds do not require, nor use, any geographic information that is available on the nodes. The information that the algorithms require is strictly derived from beacon messages during each beacon interval if the nodes are in the Power Save mode, and by overhearing other nodes if the nodes are in the powered-on backbone.

A skeleton of Odds is described as follows. An Odds node i estimates the number of neighbors it has, possibly using the sources of broadcast beacon messages in recent beacon intervals. The node then uses the result of such estimates as an indication of local node density, which is used to compute a key parameter p_i, referred to as the *backbone probability*. If the node decides to become a backbone node, such a decision is sustained for a period of K beacon intervals. Once the node becomes a backbone node, it may overhear ongoing traffic in the air by entering the promiscuous mode. If it believes that itself is currently forwarding flows, it will increase the *backbone probability* p_i accordingly, so that it has a higher probability of participating (or *renewing its membership*) in the future backbone. Otherwise, if it is indeed idle and believes at least one neighbor is forwarding traffic, it will decrease p_i to reduce its probability of participating in the future backbone. A succession of K beacon intervals is referred to as the *backbone interval*, since the structure of the backbone remains stable during such an interval. At the beginning of the next backbone interval, each node will make its own local decisions again on whether to join the backbone or to conserve energy by functioning in the Power Save mode.

B. Computation of Backbone Nodes

The purpose of computing a virtual backbone is two-fold. First, having a backbone reduces the setup latency for new connections. This is intuitive since initial route discovery broadcast messages may travel much faster on the backbone than off the backbone (operating in the Power Save mode). Second, as related work also pointed out, having a backbone is beneficial for aggregating traffic routing tasks to a selected few of nodes, which reduces contention at the MAC layer from different neighbors of the same neighborhood.

We first consider a simple, but yet effective, approach for any node to decide if it should join the backbone at the beginning of the next backbone interval. Assuming that the backbone probability p_i is known for node i. A node i participates in the backbone and enters the powered-on mode with such p_i, and such a decision is sustained for the duration of the backbone interval.

From a temporal point of view, such a series of on/off decisions on each node form a *binomial process* with probability p_i in discrete time, the length of each time slot is exactly the backbone interval (K beacon intervals). Such a binomial process is known to satisfy one key property that, if we let u_i be the number of slots between the ith and $i-1$st arrival to the system, the distribution of u_i, $i = 1, 2, \ldots$, is *geometric* with parameter p_i. Such a property holds based on the assumption that each slot contains an arrival independent of all other slots with the same probability. This assumption generally holds in our context. Therefore, any adjustments of backbone probability p_i directly affects the inter-arrival time between two arrivals, which, in our context, corresponds to the level of energy consumption on each of the nodes. Furthermore, intuitively and according to the above analysis, a node is expected to be powered on for one backbone interval during every $1/p_i$ backbone intervals, and to enter Power Save mode during the other backbone intervals.

From a spatial point of view, it may be easily derived that, the number of nodes that are powered on in the entire network during each backbone interval is *geometrically distributed*, and $\sum_{i \in N} p_i$ nodes (N is the set of all nodes deployed) are expected to be powered on while the other nodes are in PSM. The set of nodes that are powered on simultaneously forms the *virtual backbone*, to serve our purposes of fast propagation of route discovery messages, as well as traffic aggregation. The topology of this backbone changes once every backbone interval, when each node recomputes its backbone probability p_i. As such, p_i directly affects the degree of energy conservation and the connectivity of the backbone. The intuitive rules for choosing p_i are: (1) p_i should be higher where node density is low, and lower when density is high; (2) p_i should be higher on an active route when the node is actively forwarding traffic, and lower at nodes close to an active route; and (3) the set of nodes that forms the backbone at any given time should roughly form a covering set of all nodes in the network.

C. Backbone selection based on network density

The intuitive approach of choosing the backbone probability p_i is to set it to be inversely proportional to the local node density d_i around node i. This ensures that the number of backbone nodes will remain at approximately the same level.

Estimating the number of neighbors

We begin by using the number of nodes that reside within i's transmission range, i.e., the number of i's neighbors, as an indication of d_i. To estimate the actual number of neighbors n_i that it has, node i records the number of unique node identifiers (henceforth denoted by m_i) that it receives in the *beacon messages* sent during the last w beacon intervals. Here, w is an adjustable parameter that defines the monitoring window size. At the first glance of the problem, one might argue that the node i may extract source identifiers by overhearing ongoing *data* traffic in the air. This is not true, since most of the nodes we consider are in the Power Save mode, when it is not able to overhear any packets outside of the ATIM window. The only option is to estimate n_i based on received beacon messages.

Next, we estimate n_i, based on the observation that m_i nodes were heard during w beacon intervals. If a very large w is used, the number of nodes being heard will be equal to the number of nodes present with a very high probability. However, this approach only works for static or semi-static networks, since nodes heard w periods ago may have already moved out of the listener's transmission range in a high mobility scenario. Therefore, it is important to estimate n_i promptly with a smaller w.

We now analyze the relationship of w, m_i and n_i. It turns out that, in order to know $P(n_i|m_i)$ to derive $E(n_i)$, it's crucial to compute the inverse distribution $P(m_i|n_i)$ first. Note that

$$\begin{aligned} p(m_i|n_i) &= \frac{\binom{n_i}{m_i} S(m_i, w)}{n_i^w} \\ &= \frac{\binom{n_i}{m_i}}{n_i^w}\Big(m_i^w - \binom{m_i}{1} S(1,w) - \binom{m_i}{2} S(2,w) \\ &\quad - \ldots - \binom{m_i}{m_i - 1} S(m_i - 1, w)\Big) \\ &= \frac{\binom{n_i}{m_i}}{n_i^w}\Big(m_i^w - \binom{m_i}{1}(m_i - 1)^w \\ &\quad + \binom{m_i}{2}(m_i - 2)^w - \ldots \binom{m_i}{m_i - 1}\Big) \\ &= \frac{\binom{n_i}{m_i}}{n_i^w} \sum_{k=0}^{m_i - 1} (-1)^k \binom{m_i}{k} (m_i - k)^w \qquad (1) \end{aligned}$$

where $S(m_i, w)$ is the number of different lists of w node identifiers that can be constructed from m_i node identifiers, with the constraint that each of the m_i identifiers must appear in the list at least once.

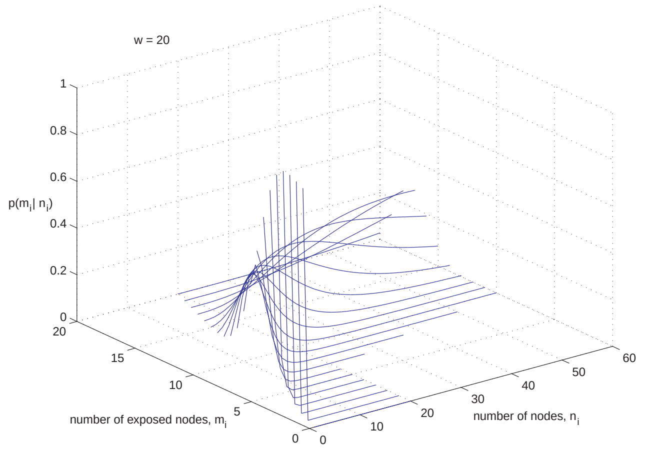

Fig. 1. Distribution of $p(m_i|n_i)$

Fig. 1 shows the distribution of $p(m_i|n_i)$, with w fixed at 20. When m_i is significantly less than w ($m_i < \frac{w}{4}$), the value of $p(m_i|n_i)$ is very large at $n_i = m_i$, and drops to almost 0 immediately as n increases. Therefore it is appropriate to estimate n_i as m_i in this case. For medium values of m_i ($\frac{w}{4} < m_i < \frac{w}{2}$), the high $p(m_i|n_i)$ values center at a small range of n_i values. Therefore estimating n_i using Eq. (2) is still appropriate. For m_i values that are larger ($m_i > \frac{w}{2}$), the non-zero values of $p(m_i|n_i)$ spread over a wide range of n_i; it is still possible to estimate n_i using Eq. (2), but the result will be of low fidelity.

Now that we have computed $P(m_i|n_i)$, we proceed to compute $P(n_i|m_i)$. It's immediate that $P(n_i|m_i) = 0$ for $n_i < m_i$. For $n_i \geq m_i$:

$$\begin{aligned} p(n_i|m_i) &= \frac{p(m_i|n_i)p(n_i)}{p(m_i)} \\ &= \frac{p(m_i|n_i)p(n_i)}{\sum_{n_i=m_i}^{N}(p(m_i|n_i)p(n_i))},\ n_i \geq m_i \end{aligned}$$

Where N is the maximum number of neighbors a node is likely to have, which, for example, can be set to be the network size. The expected number of neighbors, $E(n_i)$, can then be computed according to its definition:

$$E(n_i) = \sum_{n_i=m_i}^{N} (n_i\ p(n_i|m_i)) \quad (2)$$

Now only $P(n_i)$ is unknown in the above equations. A good modeling of $P(n_i)$ depends on specific features of the network, such as network type, network size and deployment details. For example, if approximate knowledge on average network density ρ is available, $P(n_i)$ can be modeled as a Poisson variable with parameter $\lambda = \rho - 1$. If network size N is known, $P(n_i)$ can be modeled as a uniform variable or bell-curve variable on $[0, N]$. Details here are orthogonal to our work and are not our focus in this paper.

Our estimation of n_i does not rely on overhearing data or control packets, and may therefore work well in both lightly and heavily loaded networks. However, if during the monitoring window, node i does detect through overhearing the existence of b_i additional nodes that are not in the set of the m_i beacon senders, it can adjust the values of $p(n_i|m_i)$ and $E(n_i)$ accordingly:

$$p'(n_i|m_i) = p(n_i|m_i)\frac{1}{1 - \sum_{n_i=m_i}^{m_i+b_i} p(n_i|m_i)}, n_i \geq m_i+b_i \quad (3)$$

$$E'(n_i) = \sum_{n_i=m_i+b_i}^{N} (n_i\ p'(n_i|m_i)) \quad (4)$$

Probability Redistribution

There exists one drawback of directly using the estimated number of neighbors of the node i, n_i, as its local node density d_i. Nodes that have a larger number of neighbors, which are usually more "critical" in constructing a backbone, have less chances of being selected. Fig. 2 shows a simple topology that illustrates this problem. In the figure, nodes 1 to 6, that reside on the border, each has three neighbors; and node 7, that may reach all other nodes, has six neighbors. Intuitively, the backbone that consists of exactly one node, node 7, is more efficient than the one that consists of nodes 1 to 6. However, if we simply set p_i to be inversely proportional to n_i, then p_{1-6} will be twice as large as p_7.

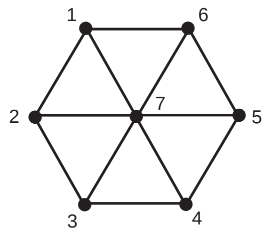

Fig. 2. A simple network of critical and non-critical nodes

The generated backbone may be improved if these more critical nodes, rather than their neighbors that connect to fewer nodes, have a higher probability of being included. Meanwhile, we still wish to preserve the property that the probability of a node being on is inversely proportional to its local node density, in order to keep the number of backbone nodes at the relatively same level. We have designed an algorithm, referred to as *probability redistribution*, that achieves both goals. In probability redistribution, a node i will first set p_i inversely proportional to the average number of neighbors ($\overline{n}_i$) that its neighbors and itself have, and then adjust this value by comparing n_i with $\overline{n}_i$ — increase the value of p_i if n_i is larger than $\overline{n}_i$, and vice versa.

In order to compute $\overline{n}_i$, node i needs to know its neighbors' number of neighbors. Therefore, we append one additional field to the beacon header of 802.11 PSM: an integer that represents the number of neighbors of the beacon sender. Upon receiving a beacon message, node i records both the node identifier and number of neighbors of the successful beacon sender. It then follows that

$$\overline{n}_i = \frac{1}{n_i + 1} \sum_{j \in \{i\} \cup N_i} n_j$$

where N_i denotes the set of neighbors of node i, and

$$p_i = \frac{c}{\overline{n}_i} \frac{n_i}{\overline{n}_i} = \frac{cn_i}{\overline{n}_i^2} \quad (5)$$

where c is a constant parameter that is tunable, and if $p_i \geq 1$, let $p_i = 1$.

For an example, in Fig. 2, after probability redistribution, p_{1-6} is proportional to $cn_1/\overline{n}_1^2 = 0.21$, and p_7 is proportional to $cn_7/\overline{n}_7^2 = 0.51$. Therefore, p_7 will become more than twice as large as p_{1-6}. After probability redistribution, $\sum_{i=1}^{7} p_i$, which in general is proportional to the number of nodes in the backbone, drops from 2.17 to 1.79. This also agrees with our observation that a more efficient backbone contains fewer nodes.

Fig. 3 and 4 shows a backbone generated from 200 wireless nodes deployed in a $1000\text{m} \times 1000\text{m}$ square within the ns-2 network simulator, with and without probability redistribution, respectively. As is evident from the illustrations, the backbone in Fig. 4 contains fewer nodes than the one in Fig. 3, and its quality is no worse than the one in Fig. 3.

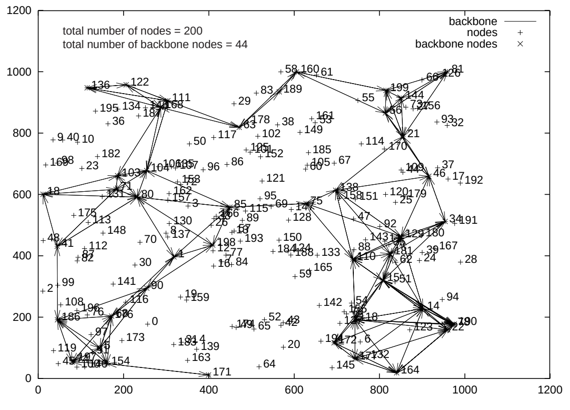

Fig. 3. Backbone generated without probability redistribution

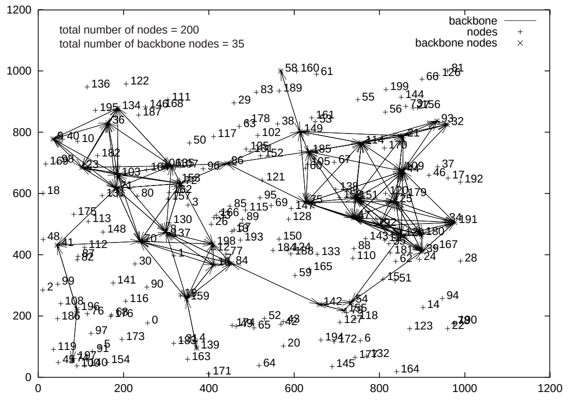

Fig. 4. Backbone generated with probability redistribution

Estimating the number of backbone nodes

The number of backbone nodes, denoted by a, increases as c increases. Below we show that a can be estimated as a function of c and p_{nb}, where p_{nb} is the probability that a pair of arbitrary nodes within the square are neighbors. Such an estimation is useful when tuning c to reach a desired backbone size. Assume the total number of nodes deployed is x, then the expected value of a is

$$E(a) \;=\; \sum_{i=1}^{x} p_i \;\approx\; \sum_{i=1}^{x} \frac{c}{n_i} \;\approx\; x\frac{c}{E(n)}$$

$$\approx\; x\frac{c}{p_{nb}(x-1)} \;\approx\; \frac{c}{p_{nb}} \qquad (6)$$

The value of p_{nb} is equal to the expected percentage of the area of deployment covered by an arbitrary node within it. Detailed computation is omitted here since they are irrelevant to our topic. We only point out that “edge effect” need to be considered in such a computation.

The result in Eq. (6) suggests that the expected number of nodes on the backbone is proportional to c, and remains at the same level as the number of nodes being deployed changes. Fig. 5 shows the number of backbone nodes for different values of c computed using Eq. (6), when $l = 1000m$ and $r = 250m$. Also listed are the average number of nodes in the backbones generated in our simulation. The value of a computed according to our analysis is close to, but slightly smaller than the ones obtained in simulation. In addition, the values of a obtained from simulation are not strictly constant, it increases slowly as x, the total number of nodes deployed increases. This is because that, in our analysis, we approximate $\sum_{i=1}^{x} \frac{c}{n_i}$ with $x\frac{c}{E(n)}$, this is clearly an underestimate.

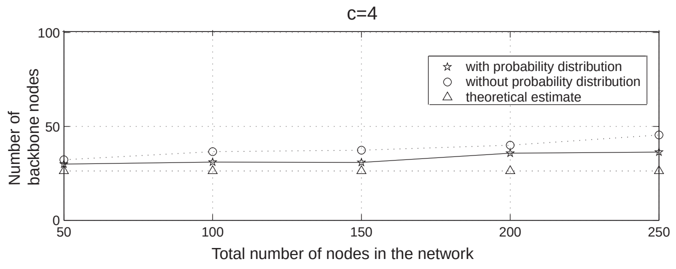

Fig. 5. Number of backbone nodes selected

Simulation results in ns-2 show that, for $c \geq 4.0$, the total number of connected components formed by the backbone nodes is very small. For 200 nodes deployed in a $1000\text{m}\times1000\text{m}$ square scenario, the average number of components we have obtained is 1.89. Furthermore, the probability that the coverage of the backbone nodes spans the entire network is very high. Assuming that the original nodes are uniformly deployed, the values of n_i and $\overline{n}_i$ at each node will be the same (except for nodes near the border), and the value of p_i will the same with or without probability redistribution. Therefore, the probability that at least one node in $Nb(i)$ is a backbone node at a given time is:

$$p = \begin{cases} 1 & n_i \leq c \\ 1-(1-\frac{c}{n_i})^{n_i} & n_i > c \end{cases}$$

Since $1-(1-\frac{c}{n_i})^{n_i}$ monotonically approaches $1-e^{-c}$ as n_i increases, we have $p \in [1-e^{-c}, 1]$. For a typical value of c in our simulations, $c = 4.0$, $p \in [0.98, 1]$. This is also verified by our simulation results.

D. Improving Average End-to-End Latency

Without further improvement, the average end-to-end packet latency using Odds will still be on the magnitude of $\frac{1}{2}kT$, similar to 802.11 PSM that we have analyzed previously. This is due to two reasons. First, as we explained earlier, with current 802.11 PSM, a packet that arrives between two ATIM windows has to wait until the next beacon interval to be marked before it can be transmitted. Second, so far our procedure of choosing p_i does not take traffic flow into consideration; nodes currently serving a flow may switch into Power Save mode, which will force packets to be rerouted.

Duplicate Forwarding

To address the first issue, we have modified the packet transmission mechanism of 802.11 PSM. When a unicast packet arrives at a node during two ATIM windows, that node can *attempt to send the packet immediately*, in the same beacon interval. Up to three attempts of retries are allowed if the first transmission fails. If the packet is sent out successfully within three retry attempts, the sending procedure terminates. Otherwise, the packet will be buffered until the next ATIM window to be advertised. With this modification, a packet can be relayed immediately to the downstream neighbor, if the neighbor is a backbone node during that beacon interval; if the downstream neighbor is a regular node in PSM, the packet will be buffered for half a beacon interval in average, as in 802.11 PSM.

Broadcast packets (e.g. route request messages from DSR) are treated differently. If a broadcast packet arrives between two ATIM windows, it will be broadcast twice: once immediately in the current beacon interval, and once in the following beacon interval (with an advertisement sent in the ATIM window first). The purpose of duplicate broadcasting is to guarantee that in case the source and destination nodes are partitioned in the Odds backbone but not in the ad hoc network topology, route request messages from on-demand routing protocols can still reach the destination.

It is desirable to introduce two further optimizations to improve the above basic *duplicate broadcast* approach. First, a node that hears the broadcast in the first time can ignore the advertisement for the same packet during the following ATIM window. Second, as proposed in [3], A node can count the number of broadcast advertisements sent during an ATIM window, and after the same number of broadcasts have been received during that beacon interval, it may go to sleep early without staying on throughout the whole beacon interval, given that there is no other unfinished unicast transmissions.

Fidelity of Ongoing Flows

To solve the second problem, we adjust the backbone probabilities p_i for nodes along an active route, such that a successive packet that is temporally not far apart from its predecessor will arrive at a backbone node with a high probability, in which case the immediate attempt in our unicast packet forwarding mechanism will relay the packet further without waiting for half of a beacon interval on average. More specifically, after the backbone probability p_i is determined according to the local network density, it will be adjusted according to local flow status before being used to make a local decision on its power status for the upcoming backbone interval.

To achieve this, a node i records the time Δt past since it last received a packet, and based on Δt, it computes the *fidelity* $q_i \in [0, 1]$ — that it believes a flow is currently being forwarded — along a *quadratic curve* as shown in Fig. 6. Therefore, node i will adjust p_i to $\max(p_i, q_i)$. The effect is that if there were packets received in the near past when p_i is being updated, the backbone probability that node i will be powered on during the next backbone interval will be high.

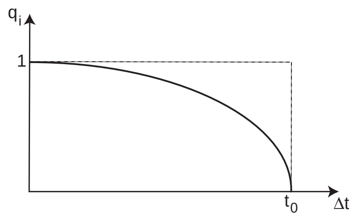

Fig. 6. Fidelity of Ongoing Flows

Neighborhood Compensation

Since nodes along an active route are expected to remain in the backbone during the ongoing session, the local density of backbone nodes along an active route is increased. Odds decrease backbone probabilities p_i at neighbor nodes along an active route to compensate for such an increase, and hence are able to keep the total number of nodes in the backbone roughly constant.

There are three ways that a node i can learn about an ongoing flow beside itself: (1) overhearing of RTS, CTS, DATA and ACK messages during beacon intervals when i is powered on; (2) overhearing of packet advertisement and ACK messages during ATIM windows; and (3) within each beacon message, add one bit that can be used to represent the current power state of the beacon sender. This bit may be set to 1 when the beacon sender is highly probable ($q_i > q_{th}$ for a threshold value q_{th}) to be involved in a flow; and set to 0 otherwise. However, exact probability compensation is hard to achieve. This is due to the fact that a neighboring node does not have complete information on the p_i, q_i and n_i values of the node on the flow being compensated. It is also because that one node may be neighbors of more than one nodes belonging to a flow, as well as nodes on more than one flows. In current simulation, we simply decrease p_i by $\frac{1-p_i}{2n_i}$ for every active neighbor observed during the last K beacon intervals. We plan to examine more precise probability compensation in future work.

E. Further discussions and insights

Odds parameters, c and K, are all tunable within a range of values. We now discuss how the choices of these parameters will affect the performance of an Odds network.

c — Recall that the number of nodes in an Odds backbone is proportional to c. Therefore, the smaller c is, the fewer nodes will be powered on simultaneously, and the less overhead is paid (with respect to energy dissipation) to maintain the backbone. On the other hand, a large c can improve the quality of the backbone, in terms of coverage and fidelity of connectivity. However, in dense networks, a large c will power more nodes on than necessary, and hence introduce high levels of interference as well as higher power consumption rates. For a very small c, Odds degenerates into a scheme that ignores setup latency for new connections, and only adapts to traffic distribution. For a very large c, Odds degenerates into an always-on network.

K — The parameter K defines the number of sustained beacon intervals each decision on p_i is applied to, and hence determines the changing rate of Odds backbone. A small K improves load balancing. A large K induces less overhead, and is desirable for setup latency control, since the probability is larger that the round trip time from sending out a route request to receiving a route reply is enclosed within one backbone interval.

We finally make an additional note with respect to load balancing. Since the Odds backbone changes every K beacon intervals, load balancing is achieved naturally and automatically. However, it is possible that a long-lived flow keeps nodes along a specific route powered-on for a long time. In that case, energy on those nodes will be exhausted sooner. It is in some sense a problem introduced by our own algorithms, since we forced a route to remain active as long as there exists an active flow, favoring end-to-end packet latency than load balancing. This is again a trade-off, since ignoring the ongoing traffic and shutting down nodes on the route will cause severe routing problems or long latencies, which in turn triggers route discovery, thus wasting bandwidth as well as introducing overhead in terms of both delay and power consumption. Another technique that may lead to better load balancing is to take the amount of power remaining at each node into consideration when deciding p_i. From later simulation results, we are convinced that with current algorithms, load balancing is not a serious issue in the long term.

IV. Performance Evaluation

We have implemented Odds within the ns-2 2.1b8a network simulator, and have performed extensive simulations to evaluate the correctness, effectiveness and performance of Odds under various network scenarios. Odds are positioned and implemented as an extension to the pure IEEE 802.11 PSM, so that they provide well defined interfaces — identical to the IEEE 802.11 standard — to the upper layers. As such, no modifications to the routing protocols are necessary or implemented. For routing purposes, we have chosen the unmodified implementation of Dynamic Source Routing (DSR) throughout our simulations, and we believe that other routing protocols (e.g. AODV, GPSR, and TORA) will also work well without modifications.

In all simulations, we use the Lucent 914MHz WaveLAN radios with 2Mbps of bandwidth and 250 meters of transmission range. With respect to the energy model, we adopt the standard energy model implemented in the wireless physical layer of ns-2 2.1b8a: no power dissipation for nodes that are powered off, 1.15 W drained power for idling nodes, 1.2 W drained power for packet reception, and 1.6 W drained power for transmission. No error models are used, and the node density is sufficient to prevent network partitioning. We use both *Power Save* and standard modes of the IEEE 802.11 MAC as the control for our comparisons, and we compare with Span when evaluating protocol scalability. Rather than distinguishing source/destination nodes from relaying nodes (as in Span or GAF), we believe it is more realistic to treat all nodes as peers in the network, which can send, receive and forward traffic and any time. All simulations are executed for 300 seconds within ns-2.

Within IEEE 802.11 PSM, we use a beacon interval of 0.2 seconds, and an ATIM window of 0.04 seconds. Within Odds, we set c in Eq. 5 as 4.0, and K as 20, which is equivalent to a backbone interval of 4 seconds given our beacon interval. We set $w = 20$ and $t_0 = 1.8s$.

A. Scalability to high node densities

We have analyzed the scalability of Odds. We have concluded that, as node density increases, Odds can scale well, keeping approximately the same number of backbone nodes. A constant backbone size leads to a reduced ratio of backbone vs. regular nodes as node density goes much higher, which amounts to better energy conservation to more nodes. In order to illustrate the validity of this claim, we conduct ns-2 experiments with 32 to 444 stationary nodes, the locations of which are randomly generated in a $600 * 600$ m^2 area[3]. With respect to traffic, we introduce three CBR connections with identical sending rates of 5 KBytes/sec. In this particular experiment, we are concerned with the average energy consumed (presented as a percentage of initial energy levels).

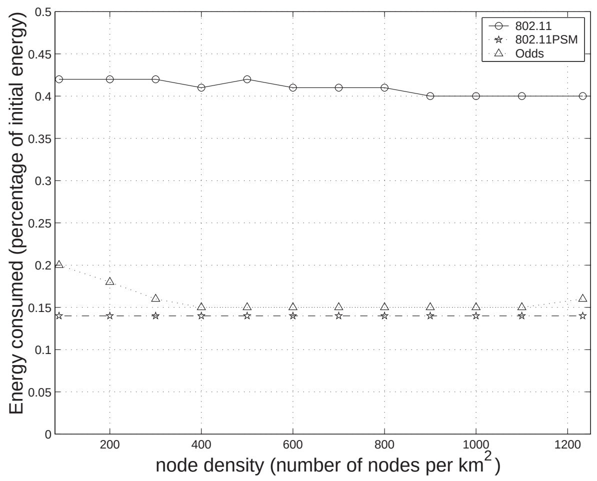

Fig. 7. Average energy consumed per node vs. node density

Fig. 7 illustrates our experimental results. We use node density (in terms of total number of nodes per square kilometer) as the x axis of the graph. This experiment puts all three candidates in the scenario where there is no or light traffic load in the network (under which Utopia consumes no or very low energy). As is evident in the figure, ranging from a relatively sparse to a extremely dense network, all three candidates – regular 802.11, 802.11 PSM and Odds — can maintain approximately the same energy dissipation rate for each node. This property can be easily achieved by 802.11 and 802.11 PSM, since with the relatively light traffic load we have injected into the network, both candidates have no problems forwarding the traffic. In this case, 802.11 PSM will roughly consume at about one third of the consumption rate of an always-on network, since it powers itself on only in the ATIM window if there is no traffic to be forwarded. The behavior of both PSM and 802.11 stays the same regardless of the node density. In contrast, this achievement is noteworthy for Odds. Since Odds are very light-weight algorithms with respect to overhead, the variations in node densities only affect the number of backbone nodes generated, which has been illustrated to approximately remain constant. This leads to a near-constant per-node energy consumption rate.

[3]The reason for a smaller region of simulation is that, to maintain the same node density in a larger region, additional nodes are required. Since ns-2 is not scalable to more than 500 nodes, we have to resort to a smaller area.

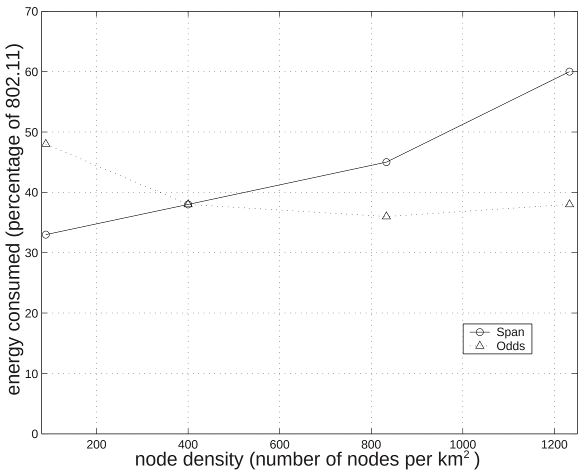

Fig. 8. Odds vs. Span: Scalability to high node densities

Although it is hard to compare the scalability of Odds with Span or GAF under identical scenarios (mainly because both distinguish between relaying and end-point nodes), we have experimented with Span under similar scenarios, e.g., identical energy models, a $600 * 600$ m^2 area and a variable node density. As illustrated in Fig. 8, we have discovered that Span works well under low densities as expected, and since it guarantees the high quality of backbone nodes, it may conserve slightly less energy than Odds with the trade-off of additional bandwidth overhead. Under higher densities, beyond those that have been attempted previously with Span, we discover that Span's performance gradually becomes poorer because the increased amount of HELLO messages per local neighborhood tends to keep nodes awake for longer periods of time. Note that this illustration does not show the bandwidth overhead of HELLO messages that may interfere with ongoing data traffic under heavier loads.

From this experiment and other experiments with different traffic loads, it is evident that Odds can achieve between one third to half of the energy consumption rate compared with a 802.11 always-on network, which is similar to Span. However, Odds are able to achieve such performance without periodic message exchanges, and can achieve approximately the same levels of performance even when nodes are extremely dense.

B. Loss rate and energy efficiency with variable offered load

We evaluate the ability of Odds to approach the performance of an always-on network with respect to forwarding heavier traffic loads (where Utopia offers identical performance as an always-on network). We consider a $600 * 600$ m^2 area, 150 nodes, as well as three CBR connections with offered load ranging from 1 to 25 KBytes/sec[4]. Fig. 9 illustrates our results. Note that we do not compare with either Span or GAF in this scenario since, obviously, a 802.11 always-on network offers the best achievable performance, with a tradeoff of the worst energy conservation.

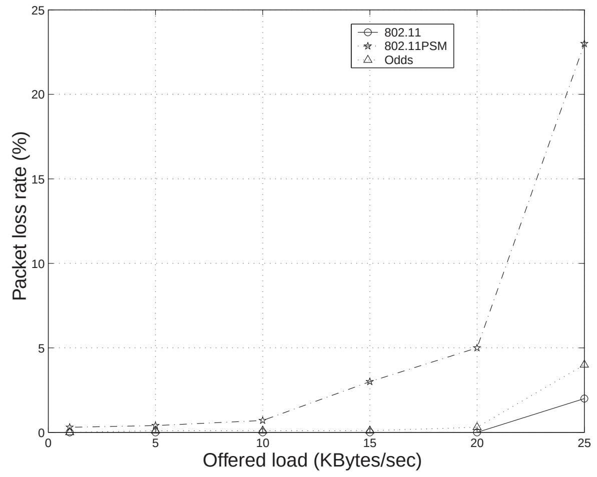

Fig. 9. Packet loss rate vs. offered load

We observe that the capacity of 802.11 PSM cannot keep up with the offered load once it becomes higher than normal. An always-on network has no unexpected problems, and Odds can closely track the performance of an always-on network, thanks to the algorithm that adapts the probability to traffic patterns.

With respect to energy conservation, we are also interested in measuring the *normalized energy consumption* under varying offer loads, defined as the amount of energy used per byte of data successfully transmitted (in the unit of Joules/byte). We assume that each node has 30J of initial energy. We note that, theoretically, 802.11 PSM should be the leader in this category, since it does not maintain a backbone. Results in Fig. 10 have confirmed this conjecture, and have further shown that Odds approach the performance of 802.11 PSM with respect to power savings. Similar performances are observed in Span as well. When the load reaches 25 KB/s, we observe that the performance of Odds even slightly surpasses 802.11 PSM, due to the much higher loss rate suffered in 802.11 PSM.

[4]Due to multi-hop interference [10], the achievable end-to-end throughput in ad hoc networks is less than $1/7$ of the channel capacity. In this scenario, this upper bound translates to about 35 KB/sec. We thus choose 25 KB/sec as a high load scenario.

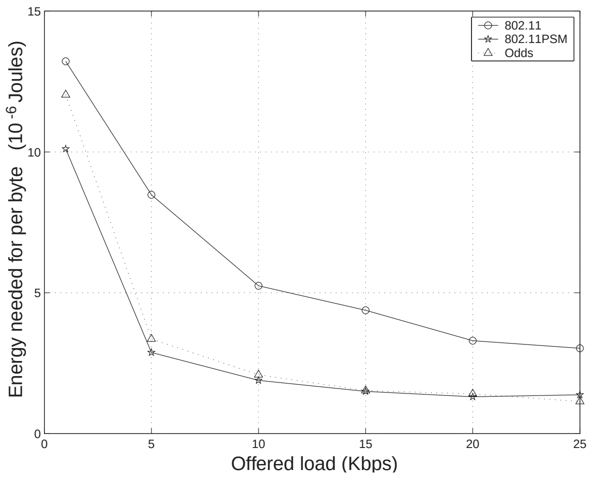

Fig. 10. Normalized energy consumption vs. offered load

C. Connection setup and average end-to-end latency

One of the major benefits of keeping a backbone in the network is to reduce the connection setup latency, so that it is scalable to the size of the network. The connection setup latency directly depends on the quality (connectivity) of the backbone. In the ideal case, if both sender and receiver are on a connected backbone, the connection setup latency should be identical to that of an always-on network.

In order to illustrate the benefits of having a backbone, we show the setup latencies of an active connection in Fig. 11, where we use one CBR connection at 5 KB/s. The results confirms the obvious observation that the leader should be a 802.11 always-on network, where setup latencies are negligible. We show that while the setup latency of Odds may not be able to compete with an always-on network (since receivers may not be on the backbone), it is an order of magnitude smaller than 802.11 PSM and converges rapidly (at the third packet it is almost the same as 802.11). Such favorable properties are not achievable without a backbone, which costs energy. It is therefore an essential tradeoff between network performance and energy conservation. Fig. 12 also shows that, compared with 802.11, Odds are able to maintain the same low levels of end-to-end packet latency, once the connection is established. In Fig. 13, we show the average end-to-end latency (including connection setup latency) under a variety of traffic loads. Due to the higher connection setup latency, the average latency of Odds may not be able to approach the level of always-on networks, but it is much lower than that of 802.11 PSM. In this category, marginally better performances are observed with Span compared with Odds in comparable network scenarios, naturally because Span maintains an always-connected backbone, while the connectivity of backbone in Odds is not guaranteed. Since Odds are able to achieve such near-optimal performance with virtually no overhead, it is still noteworthy compared with periodic message exchanges deployed in Span and GAF to maintain the backbone, just to achieve a slightly better connection setup latency.

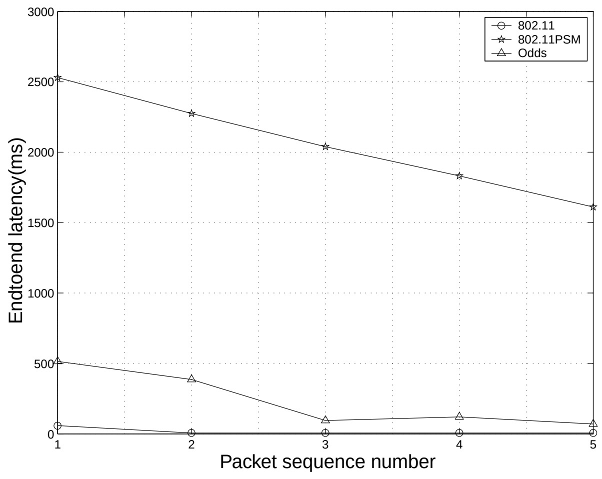

Fig. 11. The setup latency of a new connection

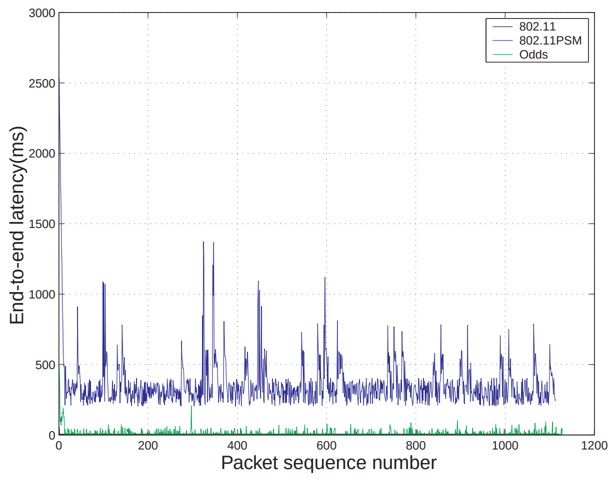

Fig. 12. The per-packet end-to-end latency in an active connection

D. Load balancing

One of the natural advantages of selecting backbone nodes based on probability is the ability to achieve load balancing without periodic overhead, compared to election algorithms in the local neighborhood. With 100 nodes and one CBR connection (5 KB/s), we show simulation results regarding the energy consumed (as a percentage of the initial energy, set at 30J) in Fig. 14 for all three candidates. As illustrated, all three candidates (802.11, PSM and Odds) can achieve almost perfect load balancing. This property is obvious for 802.11 and

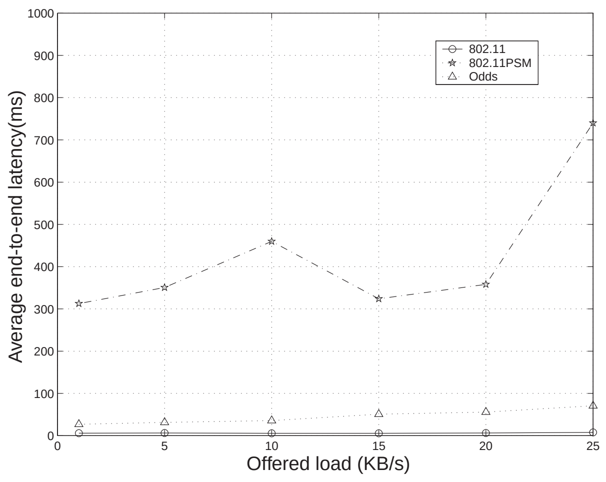

Fig. 13. Average end-to-end latency under varying offered loads

802.11 PSM, since they treat nodes as peers whose roles are identical. Since backbone-based algorithms such as Odds and Span distinguish powered-on nodes from regular nodes, it is not trivial to achieve such levels of load balancing without frequent rotation of roles (leading to message exchange overhead in Span and GAF). Odds are able to achieve this naturally with the probabilistic approach and minimum overhead.

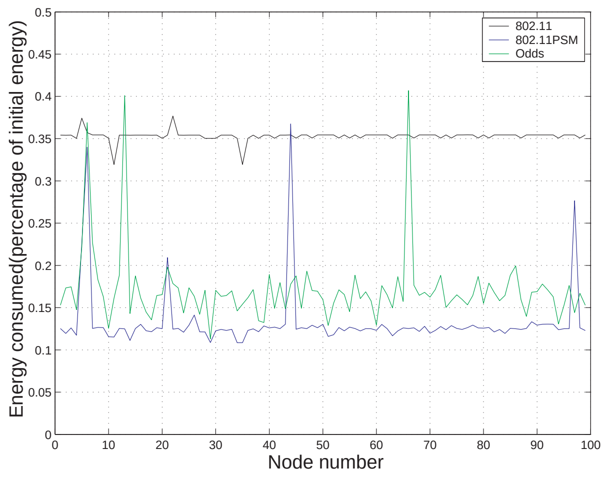

Fig. 14. Energy consumed for each node in the network, after 300 seconds

V. Conclusion

In this paper, we have presented Odds, an integrated set of algorithms to power off wireless nodes based on a probabilistic approach. Analysis and simulation results have shown that Odds have achieved the following key properties. First, they provide a scalable solution when node densities become higher, thanks to the minimum-overhead probabilistic approach that avoids periodic state exchanges in the local neighborhood. Second, they provide both a virtual backbone in order to aggregate traffic forwarding tasks and reduce connection setup latencies, and also a stable, always-on end-to-end path, adapting to the traffic patterns. Through extensive simulations in the ns-2 simulator, we are convinced that Odds have achieved our original claims, and have inherited and exceeded the current state-of-the-art. We are convinced that the full potential of Odds has yet to be realized. As future work, we will evaluate Odds by applying further mathematical analysis, and derive interesting (and possibly deterministic) properties from their seemingly random behavior.

References

[1] R. Kravets and P. Krishnan, "Power Management Techniques for Mobile Communication," in *Proc. of the Fourth International Conference on Mobile Computing and Networking (Mobicom 1998)*, October 1998.

[2] Y. Xu, J. Heidemann, and D. Estrin, "Geography-informed Energy Conservation for Ad Hoc Routing," in *Proc. of the Seventh International Conference on Mobile Computing and Networking (Mobicom 2001)*, September 2001.

[3] B. Chen, K. Jamieson, H. Balakrishnan, and R. Morris, "Span: An Energy-Efficient Coordination Algorithm for Topology Maintenance in Ad Hoc Wireless Networks," *Wireless Networks*, , no. 5, September 2002.

[4] R. Zheng and R. Kravets, "On-Demand Power Management for Ad Hoc Networks," in *Proc. of IEEE INFOCOM 2003*, April 2003.

[5] R. Zheng, J. Hou, and L. Sha, "Asynchronous Wakeup for Ad Hoc Networks," in *Proceedings of ACM MobiHOC 2003*, June 2003.

[6] B. Das and V. Bharghavan, "Routing in Ad-Hoc Networks Using Minimum Connected Dominating Sets," in *Proc. of IEEE International Conference on Communications (ICC 1997)*, 1997.

[7] D. Johnson and D. Maltz, "Dynamic Source Routing in Ad Hoc Wireless Networks," *Mobile Computing (ed. T. Imielinski and H. Korth)*, 1996.

[8] C. Perkins and E. Royer, "Ad-hoc On-Demand Distance Vector Routing," in *Proc. of 2nd IEEE Workshop on Mobile Computing Systems and Applications (WMCSA '99)*, 1999.

[9] LAN MAN Standards Committee of the IEEE Computer Society, "IEEE 802.11: Wireless LAN MAC and PHY Specifications, Chapter 11," 1999.

[10] J. Li, C. Blake, D. Couto, H. Lee, and R. Morris, "Capacity of Ad Hoc Wireless Networks," in *Proc. of the Seventh International Conference on Mobile Computing and Networking (Mobicom 2001)*, September 2001, pp. 61–69.

Zongpeng Li Zongpeng Li received his B.Engr. in 1999, from Department of Computer Science and Technology, Tsinghua University, China, and his M.S. degree in 2001 from the Department of Computer Science, University of Toronto. He is currently working towards his Ph.D. degree in the Department of Electrical and Computer Engineering, University of Toronto. His research interests include algorithm design and analysis for both wireless and wireline networks. His email address is *arcane@eecg.toronto.edu*.

Baochun Li Baochun Li received his B.Engr. degree in 1995 from Department of Computer Science and Technology, Tsinghua University, China, and his M.S. and Ph.D. degrees in 1997 and 2000 from the Department of Computer Science, University of Illinois at Urbana-Champaign. Since 2000, he has been with the Department of Electrical and Computer Engineering at the University of Toronto, where he is an Assistant Professor. In 2000, he was the recipient of the IEEE Communications Society Leonard G. Abraham Award in the Field of Communications Systems. His research interests include network-level and application-level Quality of Service provisioning, application-layer overlay networks, wireless ad hoc networks, and mobile computing. His email address is *bli@eecg.toronto.edu*.