Transport Layer Approaches for Improving Idle Energy in Challenged Sensor Networks

Yong Wang, Chieh-Yih Wan†, Margaret Martonosi, Li-Shiuan Peh

Princeton University †Intel Corporation

ABSTRACT

Today, the study of energy efficient networking solutions in sensor networks has been focusing on networks with always-on connectivity between communication end-points and short link delays. However, these assumptions are not true for networks with very long propagation delays such as Underwater Sensor Networks (UWSNs) or networks with intermittent connectivity. In such networks, idle energy expenditure, which includes energy spent on node rendezvous and idle waiting, becomes significant, and renders conventional data transport inefficient.

In this work, we leverage characteristics that are unique to such networks, i.e., long-delay tolerability and low duty-cycles, to improve idle energy efficiency. To this end, we propose a staged transport protocol, aDapTN, that adopts a store-and-forward transport paradigm with an asynchronous wakeup scheme. We evaluate the idle energy efficiency of our approach through both analysis and simulation. Our results show that aDapTN achieves much better idle energy efficiency than conventional approaches. The increased latency is a function of parameters for node rendezvous, which can be adjusted depending on the application.

Categories and Subject Descriptors

C.2.1 [**Computer-Communication Networks**]: Network Architecture and Design; C.2.2 [**Computer-Communication Networks**]: Network Protocols

General Terms

Design, Performance

Keywords

DTN, challenged sensor networks, idle energy, transport protocol, asynchronous wakeup

1. INTRODUCTION

Today, the study of energy efficient networking solutions for wireless sensor networks focuses on networks with always-on connectivity between communication end-points and short link delays. However, these assumptions are not always true. For networks with long and variable propagation delays, such as underwater sensor networks (UWSNs) or networks with intermittent connectivity, idle energy spent on node rendezvous and idle waiting for multihop communications becomes significant.

Besides, existing radios used in wireless sensor networks already consume high power in their idle mode. For example, the CC2420 radio used in MicaZ and Telos motes has three modes: the **transmit/receive mode**, **idle mode**, and **sleep mode**, consuming the lowest power in sleep mode and the most power in transmit/receive mode. When in idle mode, the radio is not communicating but the radio circuitry is still turned on which results in a ratio of idle mode to transmit mode power (-25dBM) of 1:20, as reported in [4]. This roughly 5% energy overhead for "listening" becomes significant when one considers large networks where many nodes may be listening at any time. Many other wireless interfaces also show a far-from-ideal power consumption in idle mode, as reported in [16].

Based on these observations, we argue that an efficient data transport scheme in these challenged networks should take idle energy expenditure into consideration. In this paper, we propose a new transport layer protocol that seeks to minimize network idle energy expenditure without compromising end-to-end data reliability. This is different from traditional transport layer solutions that often focus on reliable end-to-end delivery, flow control, and congestion control, etc. Our new transport protocol aDapTN, works even when the network is partitioned due to scheduled node sleep or unexpected disconnection. By dropping the assumption of end-to-end connectivity between the source and destination, significant idle energy can be saved.

Our approach is based on the Delay/Disruption Tolerant Network (DTN) architecture [5] and consists of two key components: a DTN-like store-and-forward transport paradigm for data transmission, and an asynchronous wakeup scheme for node rendezvous. Store-and-forward is used to localize data communication to a subset of nodes, creating opportunities for nodes further along the route to sleep, while an asynchronous wakeup scheme that requires no global synchronization is used to reduce the cost for node rendezvous, given that efficient time synchronization in UWSNs is still expensive to achieve [6, 20].

The store-and-forward paradigm and asynchronous node wakeup are known techniques that deal with different networking problems. Our contribution lies in the marriage of the two techniques to solve the tough problems in harsh communication environment in challenged networks. We show that DTN can be applied as an energy saving technology in a constrained environment. As far as we know, no previous work has looked at leveraging intermittent connectivity to save energy. On the contrary, other DTN variants

SIGCOMM'06 Workshops, September 11-15, 2006, Pisa, Italy.

---

[*] Other names and brands may be claimed as the property of others.

in sensor networks, such as [15], assume implicit network connectivity and adopt store-and-forward as the means to achieving high reliability.

By exploiting these two components, however, we reduce idle energy costs at the expense of data latency. For different applications, users may have different requirements regarding this tradeoff. Our approach does not enforce any specific rules and exposes such controls to the application.

To explore the relationship between traffic patterns, link delays and network diameters, we evaluate our scheme through a combination of analytical modeling and simulation. We propose analytical models for various communication models and explore their energy possibilities under different conditions. We also implement a prototype of aDapTN in TinyOS and conduct a controlled simulation study in TOSSIM [11]. Our results show that aDapTN achieves much better idle energy efficiency than conventional approaches, without compromising data delivery rate.

The main contributions of this paper are summarized as follows:

- First, we identify the constraints of existing solutions for low data rate applications in challenged sensor networks and propose a **staged** transport protocol, aDapTN, to improve idle energy efficiency.
- Second, we implement a prototype of aDapTN in TinyOS as a proof-of-concept. Simulation evaluations conducted in TOSSIM show that aDapTN can achieve from a 50% improvement in idle energy expenditure with little latency overhead, to much higher improvements at the expense of modest increases in data latency.

2. BACKGROUND

In this section, we first present two examples of challenged networks to motivate our work. They are intermittently-connected sensor networks and underwater sensor networks using acoustic communications. Then, we explore the design space of existing communication models for such networks.

2.1 Case Studies

2.1.1 Intermittently-Connected Sensor Networks

Disconnections caused by environment constraints, node failure, intentional sleep cycles, and node mobility in challenged sensor networks are not rare. BP [10] is a sensor network deployed in the North Sea aboard an operating oil tanker. The chosen oil tanker is one of the harshest environments for industrial sensor networks. The oil tanker's aft engineering spaces are constructed of steel floors and bulkheads and are divided into three major watertight compartments with hatchways in between. Sensors are spread in the compartments to perform preventive monitoring. The hatches may be periodically shut off. As a result, the sensors within that compartment will be disconnected from the base station. Since the disconnection may last for the whole night, data generated during that period need not to be sent back to the sub-cluster head immediately. Delay tolerance here can be exploited to gain energy savings.

2.1.2 Underwater Sensor Networks

There is significant research interest [3, 6, 20] in emerging applications of sensor networks to aquatic environments. Due to the unique characteristics in such environments, new challenges arise that entail re-designing some important network elements. Challenges to be overcome include severely limited link bandwidth, time-varying and long propagation delays, high bit error rates, limited battery energy that is hard to recharge, and frequent failures due to corrosion and mobility, etc. Conventional protocols are not suitable in such harsh environments.

Acoustic communications are the typical physical layer technology used in underwater networks due to the limitation of radio transmissions in underwater environment. Depending on the range of acoustic modems, the data rate can range from less than 1kbps to about 100kbps. However, to utilize high bandwidth requires limiting the distance between communicating modems to within 1km of each other, which is a significant limitation and will increase the number of hops for an end-to-end route. This renders conventional transport protocols inefficient because they require an end-to-end route established before data transmission.

These two examples motivate us to explore new communication and energy possibilities.

2.2 Communication Models

The design of energy efficient networking protocols in wireless sensor networks relies heavily on the transport paradigms and node rendezvous used. In the following, we classify communication models for UWSNs based on them.

Transport paradigms. We classify transport paradigms into two categories: multihop transmission, and store-and-forward.

In multihop transmission (**mh**), an end-to-end multihop path from the source to the sink is constructed before data transmission. A message is forwarded to the sink from a source without any delay. If a transmission to the downstream node en route fails, retransmissions are scheduled immediately to improve reliability. The message will be dropped if it cannot be delivered after a certain number of retransmissions. We define a **group** as all nodes on the path from the source to the destination, including the source and destination. The group size is defined as the total number of nodes in a group. Intra-group communication is defined as any transmission between two nodes in the group.

In store-and-forward (**sf**), a message is stored at the intermediate nodes before it is forwarded to the next hop. If disconnection happens, the forwarding node will cache the message until a connection is restored, given that there is no storage overflow. For store-and-forward, a group consists of only the communicating two nodes at any instant and the number of groups is the same as the number of hops from the source to the destination. Inter-group communication is defined as any transmission between two nodes that belong to two different groups. If we allow an k-hop subnetwork in a group, we represent it as **sfk**.

Nodes rendezvous patterns. Low-power radios and acoustic modems usually have several power modes with different power usage profile. To save energy, power management protocols are used to switch radios between different states while maintaining certain properties, such as the maximum data latency. Sleep scheduling, an important power management scheme, is often used in energy efficient MAC protocols, and sometimes in applications [18] to reduce idle listening time.

A basic problem introduced by the use of duty cycling as an energy saving technique is the need to establish rendezvous between the transmitter and receiver. Communication can only take place when the radios of both transmitter and receiver are active at the same time. Therefore, coordination is required between them so that their active time is overlapped. There are two types of rendezvous in general: synchronous and asynchronous.

In synchronous rendezvous, nodes in the network are time synchronized so that their active/sleep intervals happen at relatively the same time. S-MAC [23], IEEE 802.15.4 [2], and IEEE 802.11 Power Saving Mode (PSM) [1] are typical one-hop MAC protocols that use synchronous sleep scheduling. Multihop synchronization

Model notation	Transport		Rendezvous		Group size	Examples
	intra-group	inter-group	intra-group	inter-group		
sf-async	-	store-and-forward	-	async	2	aDapTN
mh-sync	multihop	-	sync	-	h	MintRoute, AppSleep
sfk-async	multihop	store-and-forward	sync	async	k	-

Table 1: Classification and terminology of communication models. This table is intended to be illustrative rather than definitive. h is the total hop count from the source to the destination. k is the group size that is defined as the number of hops using synchronous rendezvous. A dash indicates that a property is unavailable to that communication model.

requires at least $n-1$ pairwise synchronizations for n nodes, which is very expensive in challenged sensor networks. AppSleep [18] takes a coarse-grain approach which synchronizes all nodes on the route periodically using a SYNCH broadcast. This has been shown to be effective for low data rate stream-oriented applications. A guard time (T_{guard}) is provided to allow for clock drifts in between SYNCHs and a radio has to be awake for $2T_{\text{guard}}$ in the worst case to guarantee pair-wise rendezvous with another node.

Asynchronous rendezvous, on the other hand, allows individual nodes to wake up and sleep at different time without global coordination. Time synchronization is not needed to guarantee active time overlap between communicating pairs. However, this often comes at the cost of transmission delay. Several asynchronous rendezvous methods [21, 17] have been proposed in the literature.

3. DESIGN AND ENERGY POSSIBILITIES

In this section, we explore the communication model design space for delay tolerant challenged sensor networks and present the energy efficiency of different communication models via analytical modeling. Our energy model focuses on the relationship between idle energy consumption and various communication models, through which we present the various tradeoffs in the design space.

3.1 Assumptions

Below are the assumptions we make regarding our energy model:

Storage energy. Since ultra low power storage technology is already available [13], we do not consider storage-related operations in our energy analysis.

Network model. We assume that all sensors are connected to the sink in a multihop way. Nodes either are static or have only minor mobility if underwater networks are considered where water currents and other underwater activities can cause node movement. Since aDapTN makes few assumptions about node movement and the asynchronous wakeup scheme is agnostic to node mobility, it should also be feasible to highly mobile networks. We leave this as a future research direction. We consider a lossless channel with no packet drop caused by unreliable links during node rendezvous and data transmission. However, self-interference between downstream and upstream traffic may result in increased backoffs that reduces the actual network throughput and increase node idle waiting time in turn. Figure 1 shows a four-node line topology wherein nodes A, B or C cannot transmit at the same time due to self-interference. This reduces the actual throughput to 1/3 of that with no interference. In our analysis, we assume a constant self-interference factor $r = 3$ across the whole network for transmissions in a h hop network with $h >= 3$. For single-packet messages, such as the SYNCH packets used in synchronous rendezvous, $r = 1$.

Application. We consider a data collection application with periodic data communication from a subset of nodes to a sink that is connected to a backend server. Data rate is on the order of a few pkts/min. Messages can be a single packet or a bulk transfer with a burst of packets.

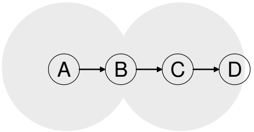
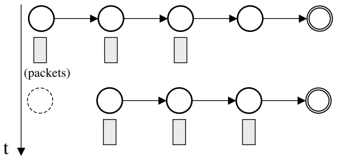
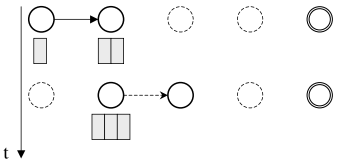


Figure 1: Illustration of self-interference in a multihop network.



(a) Multihop transmission with synchronous wakeup



(b) Store-and-forward with asynchronous wakeup

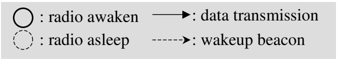


Figure 2: Illustration of different communication models. The message is transmitted in three packets and the last-hop node is the data sink. Packets are stored at intermediate nodes for sf-async, but not for mh-async.

3.2 Design Space

Table 1 shows three communication models based on discussions in Section 2.2. The models listed here are not intended to be exhaustive but they do cover a diversity of designs with very different energy/latency tradeoffs. Each model is represented as T-S with T its transport paradigm and S its rendezvous method. T can be one of the following: **sf**, **mh**, or **sfk**. S can be one of the following: **sync** for synchronous wakeup and **async** for asynchronous wakeup. A representative protocol for each model is presented. Figure 2 illustrates some of the concepts discussed here.

The first model, sf-async, is based on asynchronous rendezvous between any communication pair. It leverages the asynchrony inherent in the store-and-forward transport paradigm to work together with asynchronous wakeup schemes. Data transmission no longer requires all nodes to be powered on at the same time. A message is forwarded toward the sink as far as possible and is cached at the node where there is a disconnection, waiting for the wakeup of the next-hop node.

The second model, mh-sync, is used widely in conventional sensor networks. One example is MintRoute [22], which builds a collection tree based on the expected number of transmissions (ETX)

Parameters	Explanations	Units
D	message size	bytes
n	number of packets per message	
p	packet size	bytes
h	hop count from source to destination	
b	link data rate	bits/second (bps)
r_s	self-interference factor for SYNCH packet	
r_d	self-interference factor for data transmission	
k	number of hops per group (group size)	
g	number of groups (for sfk-async)	
T_{guard}	guard time for synchronous wakeup	seconds
T_{aw}	(worst case) asynchronous wakeup delay	seconds
T_{tx}	(worst case) per-hop packet delay	seconds

Table 2: Parameters and notations.

to the sink. Data transmission begins only after all nodes are synchronized to be awake and ready for communication. One efficient approach of achieving such rendezvous is proposed in AppSleep [18], as described in the last section.

The third model, sfk-async, is different from sf-async in that it allows synchronous and asynchronous data transfer to co-exist in the network. Nodes are organized into groups with intra-group rendezvous synchronous and inter-group rendezvous asynchronous. This model can be used to take advantage of existing short link delays in a network with varying propagation delay to achieve a low data latency, while keeping the total idle energy expenditure low.

3.3 Idle Energy and Message Latency Analysis

In our analysis, we divide idle energy into two parts: that spent on node rendezvous (E_r^{idle}) and that spent on idle waiting during data transfer (E_d^{idle}). The total idle energy for a model m is calculated as $E_m^{\text{idle}} = E_r^{\text{idle}} + E_d^{\text{idle}}$. E_r^{idle} also consists of two parts, that spent during T_{guard} and that spent during SYNCH broadcast, respectively.

In what follows, we derive the *worst case* idle energy and message latency for these three communication models, respectively. The parameters we consider include message size, packet size, hop length, link data rate, self-interference factor, etc., with their notations listed in Table 2. Since our energy model is exclusively about idle energy usage, we simply represent E_m^{idle} as E_m.

3.3.1 Store-and-forward with Asynchronous Rendezvous (sf-async)

Since each node wakes up independently using asynchronous rendezvous in sf-async, its idle waiting time is determined purely by the probability of a node stay awake, which is represented as π_w. The derivation of π_w will be presented in Section 4 wherein the implementation of a grid quorum system is described. The total awake time is then calculated as $2h\pi_w T_{\text{aw}}$ since each per-hop transfer requires two nodes to be awake for $\pi_w T_{\text{aw}}$ in the worst case. The total idle energy for sf-async, therefore, is estimated as:

$$E_{\text{sf}-\text{async}} = (2h\pi_w T_{\text{aw}})P_{\text{idle}} \tag{1}$$

Since node rendezvous is decoupled from data transfer in this model, any single message or a burst of messages will perform a rendezvous before data transfer, which incurs a maximum per-hop delay of T_{aw}. Also, packet transmissions are not pipelined and a message is forwarded to the next hop only when all packets of this message are completely received, which incurs another per-hop delay of $\frac{D}{p}T_{\text{tx}}$. Therefore, the worst case message latency is calculated as:

$$T_{\text{sf}-\text{async}} = h(T_{\text{aw}} + \frac{D}{p}T_{\text{tx}}) \tag{2}$$

3.3.2 Multihop Transmission with Synchronous Rendezvous (mh-sync)

Node rendezvous in mh-sync requires each node to be awake for at least $2T_{\text{guard}}$ to tolerate clock drifts between SYNCH packets. In a multihop network, each node also needs to stay in idle mode waiting for data packets to arrive. The duration depends on the hop count from the source to this node. Thus, the total idle waiting time spent on node rendezvous in the network is $\sum_{i=1}^{h}(2T_{\text{guard}} + r_s T_{\text{tx}} i)$ with i the hop count from the source to an intermediate node on the multihop route. Similarly, $r_d T_{\text{tx}} i$ is the worst case idle waiting time for an intermediate node i hops away from the source to receive the first data packet. Therefore, the total idle energy is calculated as:

$$\begin{aligned} E_{\text{mh}-\text{sync}} &= \sum_{i=1}^{h}(2T_{\text{guard}} + (r_s + r_d)T_{\text{tx}} i)P_{\text{idle}} \\ &= \big(2hT_{\text{guard}} + \frac{1}{2}(r_s + r_d)h(h+1)T_{\text{tx}}\big)P_{\text{idle}} \end{aligned} \tag{3}$$

In this model, node rendezvous is coupled with data transfer and is done only once before data transfer, which takes hT_{tx} to complete. Packet transmissions occur in a pipelined fashion which delivers one packet every T_{tx} once the pipeline is full. However, the pipeline is not full until the first packet reaches the destination h hops away, adding a latency of hT_{tx}, so worst case latency of the message is $(n-1+h)T_{\text{tx}}$. Since data packets are only transmitted after node rendezvous is done, we do not include the latency of node rendezvous as part of message latency and calculate it as:

$$T_{\text{mh}-\text{sync}} = r_d(n-1+h)T_{\text{tx}} \tag{4}$$

3.3.3 A Hybrid Model (sfk-async)

In this model, nodes are formed into groups: rendezvous between groups is asynchronous while that inside a group is synchronous. Since inter-group rendezvous only needs the two edge nodes from each group to be involved, the idle waiting time spent on inter-group rendezvous is only proportional to the number of groups g. Let k be the number of hops per group. Then we have $kg + (g-1) = h$ with kg the total number of hops within groups and $g-1$ the number of hops in between the g groups. Therefore, g is calculated as $\lceil \frac{h+1}{k+1} \rceil$.

The idle waiting time spent on inter-group communications can be calculated as $2g\pi_w T_{\text{aw}}$ by simply replacing h with g in Eq. (1). Similarly, the per-group idle waiting time spent on intra-group communication can be calculated by replacing h with k in Eq. (3) as $2(k+1)T_{\text{guard}} + \frac{1}{2}(r_s + r_d)k(k+1)T_{\text{tx}}$. The total idle energy is then calculated as:

$$\begin{aligned} E_{\text{sfk}-\text{async}} = \; & \big(2g\pi_w T_{\text{aw}} + g(2(k+1)T_{\text{guard}} + \\ & \frac{1}{2}(r_s + r_d)k(k+1)T_{\text{tx}})\big)P_{\text{idle}} \end{aligned} \tag{5}$$

The latency of transmissions within a group is calculated by replacing h with k in Eq. (4) as $r_d(n-1+k)T_{\text{tx}}$. The latency of inter-group communication can be treated as a sf-async model with $g-1$ virtual hops, which is calculated as $(g-1)(T_{\text{aw}} + \frac{D}{p}T_{\text{tx}})$ by replacing h with $g-1$ in Eq. (2). Therefore, the total message latency for this model is calculated as:

$$T_{\text{sfk}-\text{async}} = gr_d(n-1+k)T_{\text{tx}} + (g-1)(T_{\text{aw}} + \frac{D}{p}T_{\text{tx}}) \tag{6}$$

	CC2420	UWM1000
transmit	28.1mW (-25dBM)	2W
receive	62.1mW	0.75W
idle power (P_{idle})	1.41mW	8mW

Table 3: Reported power numbers of CC2420 and UWM1000.

3.4 Analytical Results

The energy related parameters used for our analysis is based on the CC2420 family of low-power, 802.15.4-compatible radios from Chipcon, which have been used in many sensor platforms. We use the published data from [4], as listed in Table 3. For acoustic modems, no reported empirical evaluation is available as far as we know. So we list the reported power numbers of UWM1000 [14], one widely used model, as a reference here.

Figure 3 plots the worst case idle energy expenditure with different message size and link data rate and Figure 4 plots the worse case message latency. We use two link data rates: 100kbps and 5kps, which are representative values for conventional sensor networks and UWSNs, respectively. For the 5kps case, we use a propagation delay of 0.67s that is typical for acoustic communications in UWSNs. To study the impact of single packet transfer and multi-packet bulk transfer, we use both a 512-byte message and a 30-byte message. Since node rendezvous is done once for mh-sync, we conjecture that aDapTN will gain more benefit for small messages since the rendezvous cost is amortized across all packets in mh-sync. Active energy refers to that spent on message transmissions and is plotted here to show the relative importance of idle energy expenditure. We assume lossless channels and no transmission contention. In reality, packets may be retransmitted, which produces higher active energy expenditure. However, the relative trends shown in the figures will still remain since retransmissions normally will not change the order of magnitude of energy consumption.

Comparing Figure 3(a) and Figure 3(b), we see that for networks with low data rates and long propagation delays, the active energy dominates for bulk data transfer, while idle energy dominates for small messages. However, as hop counts increase, idle energy expenditure will surpass active energy expenditure, which is illustrated in Figure 3(b). This is because the total idle energy cost grows exponentially with increased hop counts. Therefore, for long-hop networks, aDapTN will gain more benefits. In terms of idle energy efficiency, sf-async, sf4-async and sf9-async consistently outperform mh-sync because they do not require all nodes en route to be in idle mode. Comparing to sf9-async, sf4-async consistently has a lower idle energy expenditure. This is because more nodes are involved in asynchronous communications in sf4-async. However, sf9-async has a smaller worse case data latency as illustrated in Figure 4(b) because more nodes use mh-sync communications.

Figure 3(c) shows that for networks with high data rates and short propagation delays, a similar relationship between various model exists, although the difference between them are not as significant as in a more challenged network. This is because idle energy is determined by factors that grow in challenged networks. An interesting point to note is the relative energy expenditure between sf-async, sf4-async and sf9-async. As shown in Figure 3(a) and Figure 3(b), for a model with a larger group size, it will have a higher idle energy expenditure since more nodes are involved in mh-sync. However, in a network with high data rate and low propagation delays, mh-sync may incur a much smaller *per-hop* transmission latency. The smaller delay in upstream transmissions for a group-oriented model will lead to reductions in idle energy expenditure in downstream nodes. However, mh-sync still consumes the most energy among all models.

Figure 4 demonstrates that mh-sync consistently achieves the lowest data latency. Comparing sf-async and sfk-async (n=4,9), we see that sfk-async achieves significant improvement in data latency and a slightly higher idle energy usage. This comes as no surprise since sfk-async has a group size of k which results in much fewer number of inter-group hops, which is directly proportional to the worse case data latency. Further, since the idle energy expenditure for mh-sync grows exponentially with the number of hops, the idle energy increase only slightly for small group sizes. In summary, sfk-async ($k > 2$) achieves a good balance between idle energy expenditure and data latency. It can be adjusted dynamically to meet different user preference regarding the energy/latency tradeoff.

4. DESIGN AND IMPLEMENTATION

In this section, we describe the architecture of aDapTN and provide details about our prototype implementation.

4.1 Core Algorithms

As a concrete implementation of the generic communication model described in previous sections, aDapTN consists of core algorithms related to node rendezvous, routing, etc., which are explained in this section.

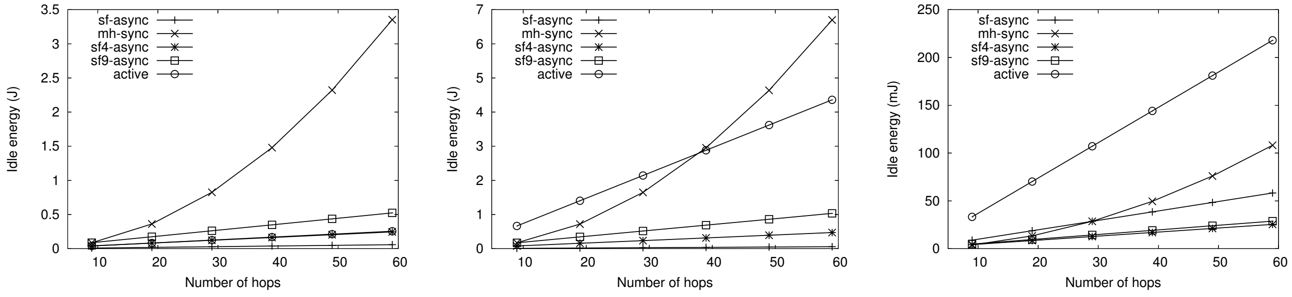
Quorum-based wakeup. We borrow the idea from [21] and use a quorum-based wakeup scheme to wake up the next-hop node in a multi-hop network. The quorum system we used is a grid quorum system with applications in many other areas, such as distributed mutual exclusion [12]. In brief, if we divide one round of schedule into q^2 time slots, the radio only needs to be powered on for $2/q$ of the schedule duration to guarantee rendezvous with another node to ensure one communicate. The selection of q is a design parameter. A higher q will result in very efficient power usage. However, it will lead to a longer delay. An example grid quorum system with $q = 4$ is illustrated in Figure 5. Each grid represents the quorum system used by one node. We call the time slots that a node needs to be awake a *quorum group* and the length of each such time slot a *quorum interval*. The radio should be either on or off during each slot and it only needs to wake up in the quorum group. The two nodes use different quorum groups shown as the shaded region in the matrix. The highlighted regions are those in which the two nodes overlap.

In each quorum interval, the node needs to send out a beacon message first for synchronizing with other potential neighbors, as illustrated in Figure 6. Once two nodes are synchronized with each other, they can keep on communicating until all buffered messages are transmitted. Then, they can resume their normal schedule independently again and wait for the next rendezvous.

If we assume the clocks of the two nodes are synchronized, it is easy to see that such a quorum system will overlap twice for one round of schedule. However, we can prove that even if their clocks are not synchronized, they can guarantee to hear each other's beacon message at least once for each round using our wakeup method. Due to space limitation, we omit the formal proof here.

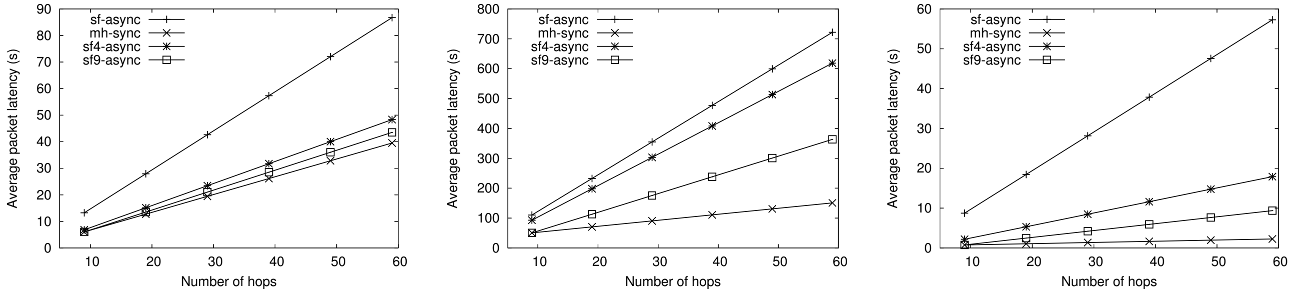
Our transport protocol can be used with any MAC layer protocol that handles the micro-level issues such as channel contention, hidden terminal problems, etc. Data collection schedule is controlled by aDapTN separately at a macro-level. This approach keeps the MAC layer simple and allows for reuse of well-understood MAC protocols.

Tree construction. For each rendezvous, aDapTN exchanges route update information and link quality estimation information as in MintRoute. Though the dissemination of routing information



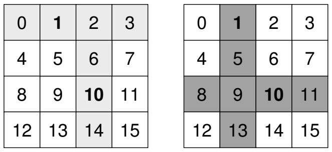
(a) $D = 30, b = 5k, T_{\text{tx}} = 0.67$.
(b) $D = 512, b = 5k, T_{\text{tx}} = 0.67$.
(c) $D = 512, b = 100k, T_{\text{tx}} = 0.01$.

Figure 3: Idle energy expenditure for different message size, link data rate and propagation delay. There are a total of 5 groups for sfk-async.



(a) $D = 30, b = 5k, T_{\text{tx}} = 0.67$.
(b) $D = 512, b = 5k, T_{\text{tx}} = 0.67$.
(c) $D = 512, b = 100k, T_{\text{tx}} = 0.01$.

Figure 4: Data latency for different message size, link data rate, and propagation delay. We use two group sizes (4 and 9) for sfk-async.



0	**1**	2	3
4	5	6	7
8	9	**10**	11
12	13	14	15

0	**1**	2	3
4	5	6	7
8	9	**10**	11
12	13	14	15

Figure 5: An example of a grid quorum system with $n = 4$. The two quorum groups overlap at time slots 1 and 10.

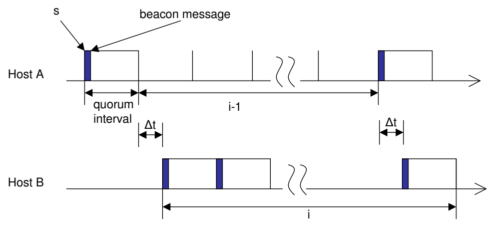


Figure 6: An example of quorum-based asynchronous wakeup.

may be slow compared to MintRoute, our protocol can still find a good parent given that every other two nodes can see each other within each schedule.

Rendezvous contention. Since the quorum group selected by each node is randomly initialized, two or more nodes could be competing to be synchronized with an awaken node. This introduces contentions during rendezvous. Even worse, beacon messages used for node rendezvous may interfere with normal traffic transmissions. To address this problem, we assign different priorities to different traffic. For those that require immediate response, we assign a high priority by setting their expiration timers to the smallest in order for them to grab the channel the first. In aDapTN, priority is assigned in decreasing order to the following types of traffic: data transmission traffic, routing update traffic and other control message exchanges. This simple scheme works very well for reducing contention and interference among different traffic.

4.2 System Architecture

This section describes our prototype implementation of aDapTN in TinyOS, with its system structure shown in Figure 7. The shaded blocks comprise the control plane, which includes the routing stack and the rendezvous manager which controls the synchronization between communicating nodes. In our current design, we choose a tree-based data collection routing protocol similar to MintRoute [22]. Other networking protocols, such as geographic routing, can also be used. The rest components comprise the data plane that work together to forward data messages. Currently, our implementation of aDapTN only supports sf-async; we are working on extending aDapTN to support sfk-async.

5. EVALUATION

To aid the development of aDapTN and to better understand its behavior and design tradeoffs, we evaluate aDapTN using simulation in this section. The simulator models the wireless channel behavior based on packet loss distribution data collected from a real-world testbed. Although not perfect, it allows us to quickly examine the performance of aDapTN. Furthermore, the simulator is originally designed for traditional sensor networks and the results presented here is applicable to challenged sensor networks of motes using CC2420. We are actively working on an appropriate simulation environment for UWSNs.

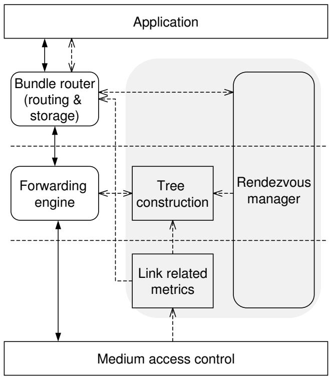


Figure 7: Component block diagram of aDapTN.

5.1 Experiment Setups

We use the TOSSIM-CC2420 simulator[1] provided in the TinyOS distribution. It models the CC2420 radio with a data rate close to 100kbps. This matches the environment of the BP application described in Section 2. TOSSIM-CC2420 incorporates PowerTOSSIM [19], a power modeling extension to TOSSIM. PowerTOSSIM can model power consumed by TinyOS applications on Mica2, MicaZ, and Telos motes.

We compare the performance of aDapTN with MintRoute, which needs to first wake up all nodes involved in the communication and then pump data to the sink in a multihop manner. Since the original design of MintRoute does not specify ways to achieve node rendezvous, we use the scheme proposed in AppSleep for such purpose and use MintRoute as the multihop transmission scheme.

We use a low-rate data collection application for our evaluations with a message size of 30 bytes that can fit into one packet in TinyOS. The message arrival rate is set at 4 pkts/min. The quorum size is set at 16 and the quorum interval is set at 1200ms. We use two topologies, a 3x3 grid and a 1x12 line topology, to create networks with different hop counts. For the 3x3 topology, node 9 is selected as the source and node 1 as the sink; for the 1x12 topology, node 12 is selected as the source and node 1 as the sink. Each experiment lasts 600s. We run each experiment 5 times and the average is shown in the results section.

5.2 Performance Metrics

We consider the following performance metrics in our evaluation.

Success rate: The fraction of messages that are delivered to the sink. Since data reliability is of high priority in many applications, our proposed scheme should have at least as high a success rate as conventional approaches. In our experiments, since both MintRoute and aDapTN achieve the same success rate for the scenarios we simulated, we omit the success rate comparison.

Average idle energy per node: The idle energy expenditure averaged among all nodes.

Average data latency: The average message latency among all successfully delivered messages.

5.3 Results

Table 4 shows the idle energy results for both MintRoute and aDapTN, which demonstrates that for both the 3x3 and 1x12 topologies, aDapTN spends much less time in idle mode to deliver the same amount of data. It achieves idle energy savings at the cost of increased latency. For the 1x3 topology wherein the hop count from the source to the sink is 3, its average latency is 75s. For the 1x12 line topology, its average latency is 228s due to increased hop counts from the source to the sink. However, the delays are within several minutes. For a typical data collection application, such delays are acceptable, given that almost half of the idle energy can be saved compared to conventional approaches.

Quorum size	Energy per node (mJ)	Average latency (s)
16	3264	75
36	2481	774

Table 5: Performance of aDapTN with different quorum size (16 vs. 36).

For networks with very low data rates and long propagation delays, our postulation is that aDapTN can gain even more benefit in terms of idle energy savings because idle waiting time saved is proportional to link delays and inversely proportional to data rates. We are currently working on the evaluation of aDaTN for simulated underwater environments.

Furthermore, if a node and its neighbors can only discover each other in an asynchronous way, the relay nodes selected may not be the optimal ones. Given that we can control the delay of information exchange using asynchronous wakeup and link quality usually will not fluctuate sharply during small time intervals, we can still have a routing structure that is close to the optimal one.

Impact of quorum size. As we introduced in Section 4, a grid-quorum system can tune its parameter q to trade in energy with data latency or vice versa. For the above experiments, we use a quorum group of 16 ($q = 4$) time slots. Therefore, during each round of schedule, the radio needs to be on for 7 time slots. For this experiment, we change the quorum group size to 36 ($q = 6$). Therefore, each node needs to be on 11 of the 36 time slots. Analytically, this will produce idle energy savings of $(7/16 - 11/36)/(7/16) = 30.2\%$.

Table 5 shows the tradeoffs using different quorum size. For the same amount of running time, using a quorum size of 36 will produce energy savings of 24.0%. This is close to the analytically estimated number. Many factors can contribute to the slight difference, such as packet retransmissions, which are not considered in the analytical model.

6. RELATED WORK

Our work is related to delay/disruption tolerant networks, asynchronous wakeup in wireless ad hoc networks, and power management in sensor networks. We discuss the most relevant work in each category in turn.

DTN related. The DTN architecture presented in [5] provides a generic network architecture for various challenged networks. The authors in [7] discuss various ways to apply the DTN architecture to sensor networks. DTNlite [15] presents a real implementation of a stripped-down version of the DTN architecture in TinyOS on resource-constrained motes. However, their approach assumes an always-on network which is not suitable for more challenged sensor networks such as UWSNs. The only work we are aware of on power management in DTN is presented in [9]. Their approach targets mobile networks and their goal is on maximizing contact opportunities between nodes when power management is used. Furthermore, their approach assumes that nodes are time synchronized which is a strong assumption in challenged sensor networks such as UWSNs. Our approach, however, can work even if node clocks are not synchronized.

---

[1] `tinyos-1.x/beta/TOSSIM-CC2420/`

	3x3 grid		1x12 line	
Communication model	**Energy per node (mJ)**	**Average latency (s)**	**Energy per node (mJ)**	**Average latency (s)**
mh-sync (MintRoute, AppSleep)	6616	7	6683	16
sf-async (aDapTN)	3264	75	3358	228

Table 4: Experimental performance using two different topologies.

Asynchronous wakeup. Tseng *et al.* [21] propose three asynchronous power management protocols for mobile ad hoc networks where synchronized power management is difficult, such as networks with unpredictable node mobility and networks with no clock synchronization mechanism. Later on, they identify in [8] a rotation closure property that allows for a more flexible quorum system design. Zheng *et al.* [24] propose an asynchronous wakeup scheme based on block combinatorics design and an on-demand power management protocol based on it.

Power management. A stream-oriented power management protocol is proposed in [18] to support a class of sensor network applications characterized by delay tolerant, asynchronous data traffic and scheduled data transmission. An application-layer wakeup/sleep scheme is proposed to enable energy-efficient network operations by only keeping the active route between a source and receiver awake. This scheme relies on the existence of a stable and fixed end-to-end route during the entire data stream transmission, which is hardly applicable in challenged networks, such as underwater environments.

7. CONCLUSIONS AND FUTURE WORK

In this paper, we have presented aDapTN, a new transport protocol based on a staged communication model that saves significant idle energy in challenged wireless sensor networks. Our technique consists of two core components: store-and-forward transport and asynchronous rendezvous. It saves idle energy by relaxing the requirement for end-to-end connectivity during data transmission and allowing the network to be disconnected intermittently via scheduled sleeping. As far as we know, no previous work has leveraged DTN techniques to save energy in such environments. Although our case studies are centered on underwater sensor networks and intermittently-connected sensor networks, we expect our approach to be useful to other challenged networks wherein idle energy efficiency is crucial.

Due to the limitations of our experiment environments, we have not fully evaluated the feasibility of aDapTN in UWSN settings with varying, long propagation delays and high packet loss rates, etc. We expect that aDapTN should perform even better in such environments due to its robustness to such challenges. We are working on revising our current simulation environment to reflect those real-world constraints and anticipate a more systematic evaluation based on it. Another avenue for future work is an adaptive transport protocol that can dynamically switch between different communication models when situation changes. The group-oriented model is one such protocol. Efficient design of such a protocol is an area of future work.

Overall, we feel this initial work already offers significant opportunities for saving idle energy, and further work has high potential for improving this.

8. REFERENCES

[1] *IEEE Standard 802.11: Wireless LAN Medium Access Control and Physical Layer Specifications*. 1999.

[2] *IEEE Standard 802, part 15.4: Wireless Medium Access Control (MAC) and Physical Layer (PHY) Specifications for Low Rate Wireless Personal Area Networks (LR-WPANs)*. 2003.

[3] I. F. Akyildiz, D. Pompili, and T. Melodia. Underwater acoustic sensor networks: research challenges. *Elsevier's Journal of Ad Hoc Networks*, 3(3):257–279, 2006.

[4] Chipcon CC2420 2.4 GHz IEEE 802.15.4 / ZigBee-ready RF Transceiver. http://www.chipcon.com/.

[5] K. Fall. A delay-tolerant network architecture for challenged Internets. In *Proc. ACM SIGCOMM*, 2003.

[6] J. Heidemann, W. Ye, J. Wills, A. Syed, and Y. Li. Research challenges and applications for underwater sensor networking. In *Proc. IEEE WCNC*, 2006.

[7] M. Ho and K. Fall. Poster: Delay tolerant networking for sensor networks. In *IEEE SECON*, 2004.

[8] J.-R. Jiang, Y.-C. Tseng, C.-S. Hsu, and T.-H. Lai. Quorum-based asynchronous power-saving protocols for IEEE 802.11 ad hoc networks. *Mobile Networks and Applications*, 10(1-2), 2005.

[9] H. Jun, M. H. Ammar, and E. W. Zegura. Power management in delay tolerant networks: A framework and knowledge-based mechanisms. In *Proc. IEEE SECON*, 2005.

[10] L. Krishnamurthy, R. Adler, P. Buonadonna, J. Chhabra, M. Flanigan, N. Kushalnagar, L. Nachman, and M. Yarvis. Design and deployment of industrial sensor networks: Experiences from the north sea and a semiconductor plant. In *Proc. ACM SenSys*, 2005.

[11] P. Levis, N. Lee, M. Welsh, and D. Culler. TOSSIM: Accurate and scalable simulation of entire TinyOS applications. In *Proc. ACM SenSys*, 2003.

[12] M. Maekawa. A $\sqrt{N}$ algorithm for mutual exclusion in decentralized systems. *ACM Transactions on Computer Systems (TOCS)*, 3(2), 1985.

[13] G. Mathur, P. Desnoyers, D. Ganesan, and P. Shenoy. Ultra-low power storage for sensor networks. In *Proc. IPSN/SPOTS*, 2006.

[14] Link-quest acoustic modem specifications. http://www.link-quest.com/.

[15] R. Patra and S. Nedevschi. DTNLite: A reliable data transfer architecture for sensor networks. Technical report, 2003. CS294-1 Course Project Report, Berkeley.

[16] T. Pering, Y. Agarwal, R. Gupta, and R. Want. CoolSpots: reducing the power consumption of wireless mobile devices with multiple radio interfaces. In *Proc. ACM MobiSys*, 2006.

[17] J. Polastre, J. Hill, and D. Culler. Versatile low power media access for wireless sensor networks. In *Proc. ACM SenSys*, 2004.

[18] N. Ramanathan, M. Yarvis, J. Chhabra, N. Kushalnagar, L. Krishnamurthy, and D. Estrin. A stream-oriented power management protocol for low duty cycle sensor network applications. In *Proc. IEEE EmNets*, 2005.

[19] V. Shnayder, M. Hempstead, B. rong Chen, G. W. Allen, and M. Welsh. Simulating the power consumption of large-scale sensor network applications. In *Proc. ACM SenSys*, 2004.

[20] A. Syed and J. Heidemann. Time synchronization for high latency acoustic networks. In *Proc. IEEE Infocom*, 2006.

[21] Y.-C. Tseng, C.-S. Hsu, and T.-Y. Hsieh. Power-saving protocols for IEEE 802.11-based multi-hop ad hoc networks. In *Proc. IEEE Infocom*, 2002.

[22] A. Woo, T. Tong, and D. Culler. Taming the underlying challenges of reliable multihop routing in sensor networks. In *Proc. ACM SenSys*, 2003.

[23] W. Ye, J. Heidemann, and D. Estrin. An energy-efficient MAC protocol for wireless sensor networks. In *Proc. IEEE Infocom*, 2002.

[24] R. Zheng, J. C. Hou, and L. Sha. Asynchronous wakeup for ad hoc networks. In *Proc. ACM MobiHoc*, 2003.