Collaborative Data Delivery in Energy-Synchronized Sensor Networks

Mingsen Xu† Wen-Zhan Song†
† Georgia State University, Atlanta GA 30303
†mxu4@student.gsu.edu, †wsong@gsu.edu

***Abstract*—In an energy-synchronized sensor network, each sensor node needs to synchronize its duty-cycle with its residual energy (or energy harvesting rate) to meet a predefined network lifetime requirement. A key research challenge of such energy-synchronized sensor networks is how to collaboratively utilize the heterogeneous node duty-cycles for efficient and fair data delivery. In the proposed system, a collaborative data delivery protocol is designed to exploit multiple energy-synchronized paths based on a new max-flow min-variance algorithm. In consort with this data delivery protocol, a localized TDMA MAC protocol is designed to synchronize nodes' duty-cycles and mitigate media access contentions. The time synchronization is realized through a low-cost and low-power WWVB receiver. We have conducted extensive simulations and experiments to evaluate the system. The results demonstrate that this design not only increases network throughput and fairness, but also increases network lifetime, comparing to the existing protocol stack in TinyOS for duty-cycled sensor network.**

***Index Terms*—Collaborative Data Delivery, Max Flow, Energy-Synchronized Sensor Network**

I. Introduction

Sensor networks have been applied to many critical applications in the past decade, however, a major concern of practice remains: can a battery-powered sensor network provide required monitoring service quality while meeting lifetime requirement? To meet network lifetime requirement with limited power supply (battery or renewable energy source), a good strategy is to put radios into periodic active and sleep mode, which is called *duty-cycle*. The node has to synchronize its resource usage to its energy supply. With lifetime requirement, sensor nodes need to plan their energy usage for each time cycle. Moreover, either residual energies or energy harvesting rates are often *heterogeneous* across a network, thus nodes' duty-cycles are heterogeneous. With nodes running in different duty-cycles, how to synchronize node energy usage (e.g., duty-cycles and Tx/Rx durations) of multiple-paths to maximize data delivery throughput remains an open research question.

Energy conservation has been the central focus since sensor networks were born. The existing techniques include MAC, routing, topology control protocols, as well as cross-layer optimizations, and have made great progress toward the goal. However, network efficiency and lifetime maximization through collaborative network resource utilization have been under explored. Sensor networks have scarce resources (energy, bandwidth, CPU, memory). To make the best and most efficient use of these resources, sensor nodes need to collaboratively utilize their resources to maximize the global data quality and network lifetime. In this paper, we propose an innovative system design where nodes collaboratively utilize the heterogenous duty-cycles to maximize network performance while meeting network lifetime constraints. The main technical contributions of this paper are:

- A collaborative data delivery protocol to maximize network throughput and fairness and synchronize nodes duty-cycles for efficient communications.
- In consort with the collaborative data delivery, a localized TDMA MAC protocol is designed, which synchronizes nodes' duty-cycles and eliminate media contention for efficient communication on the selected flow paths. For time synchronization, in our platform (TelosW [1]) a WWVB radio receiver is attached to provide millisecond synchronization accuracy with μA current consumption.
- Extensive evaluations are conducted in both real testbeds and network simulators. The results demonstrate that this design not only increases network throughput and fairness, but also increases network lifetime, comparing to an existing protocol stack of X-MAC and CTP.

We note that the max-flow and min-variance algorithm given in this paper also provide a way to estimate the best network throughput and fairness under given lifetime constraints.

The rest of the paper is organized as follows. Section II presents the problem statement and motivation. Detailed system design is presented in Section III. In section IV, we presents our system evaluation results in both real testbed and simulators. The related works are reviewed in section V. We conclude the paper in section VI.

II. Problem Statement and Motivation

In a duty-cycled sensor network, sensor node can stay in any of the three states: *transmitting*, *receiving* and *sleeping*. Given a node u, we denote T_u, R_u and S_u as its total time duration of transmitting, receiving and sleeping respectively in the entire lifetime. Clearly, the transmitting and receiving data rate of node u is proportional to the active time duration of transmitting and receiving. If link loss ratio is also considered, then T_u and R_u represent the effective transmission and receive data rate of node u. Then the node u's capacity $C_u = (T_u + R_u) \cdot r$ denotes the total flow node u can handle, where r is the radio data rate in each node. Let D_u be the bandwidth demand of node u's own data, then the flow conservation $T_u = R_u + \frac{D_u}{r}$ shall hold to avoid data losses. To meet lifetime requirement, C_u shall be synchronized with

the usable energy of each node u. Under the heterogeneous constraints on node capacity C_u, network nodes need to find T_u and R_u for each node u, such that the network throughput is maximized and flow fairness is guaranteed. We solve this through the following max-flow and min-variance algorithm.

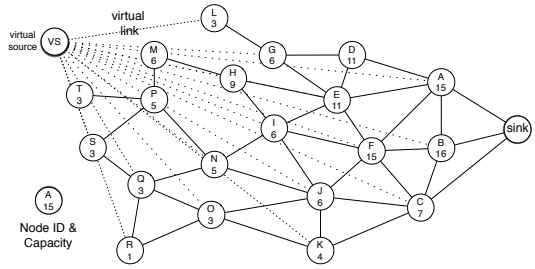


Fig. 1. Illustration of original network topology and node capacity. In a many-to-one data collection network, a Virtual Source (VS) can be added to apply max network flow algorithm.

In a heterogeneous duty-cycled sensor network, a data collection tree for data delivery is not necessarily the best solution. Each node may use multiple paths to deliver data for itself and its upstream nodes. Clearly, multiple paths may provide nodes with more delivery opportunities. In Figure 1, we show a sensor network with heterogeneous node capacities. It can be clearly observed that using collaborative "multi-path" can deliver more data than using only one single path. To be more specific, in Figure 2, we illustrate that there are 4 starved nodes if a collection tree is used. Here, we assume each node has one unit flow, which applies to the rest figures and examples in this paper. A node is starved when its own unit flow is held due to the bottleneck in the downstream nodes. In Figure 2, node T and S are starved, because their unit flows are throttled due to the bottleneck in node I. Compared with the Figure 3, it is easy to see that collaborative "multi-path" can deliver more data than a single path. And it motivates us to fully utilize the collaborative data delivery paradigm to improve data throughput. Then, the challenge is how to find the optimal paths for each individual node and how to manipulate the data flows such that network throughput and flow fairness are maximized, assuming network nodes' duty-cycles are fixed but heterogeneous. Next, the problem of finding max delivery throughput can be converted to the max flow problem. To solve the max flow problem in this converged tree topology, a virtual source is added and connected to each source node, which transforms the "many-to-one" flow maximization problem to "one-to-one" flow maximization problem.

We formally state our throughput maximization problem. Given a sensor network $G = \{V, E\}$, where V is the set of nodes and E is the set of links. Each node is subject to its energy capacity constraint: $C : V \rightarrow \mathbb{R}^+$. Further, let $f_u(v, w)$ denote the flow from node v to w originated at source node u. Our objective is to find the optimal flow scheduling: $\{f_u(v, w)\}$, such that the network throughput $(F = \sum_{x \in V \setminus sink} f_x(x, sink))$ is maximized, and the variance of originating flow for each node $(f_u(*, sink))$ is minimized.

Table I shows the notations used in the paper.

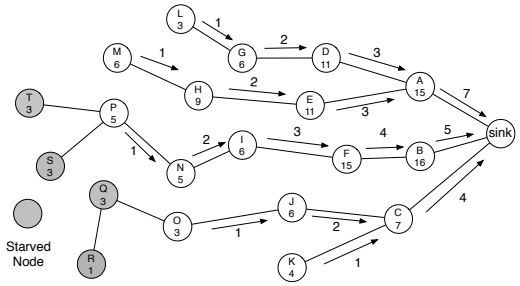


Fig. 2. Illustration of data collection tree, where 4 nodes are starved.

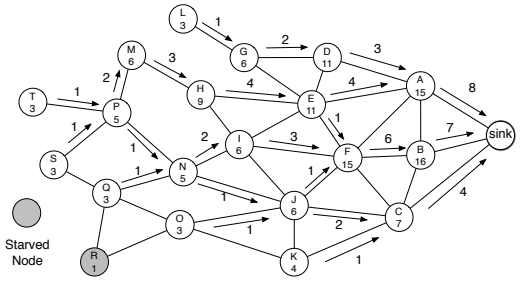


Fig. 3. Illustration of max flow network generated by our "loop-free" max-flow algorithm, where only 1 node is starved. In this and following examples, we assume each node has unit flow demand.

TABLE I
LIST OF NOTATIONS IN MAX-FLOW AND MIN-VARIANCE ALGORITHM

C_u	capacity of node u
D_u	source flow demand of node u
N_u	neighbors of node u
r	radio data rate
$h[u]$	height function of node u
E_f	edge set in the residual graph
$\delta_f(u, v)$	distance between u and v in residual graph
$C_f(u, v)$	residual capacity from node u to node v
$excess[u]$	excess flow of node u
$f_*^{max/min}$	max or min source flow in the flow network
$f_u(v, w)$	flow from node v to w with source node u

III. SYSTEM DESIGN

The proposed system design includes both network and MAC layer protocols. In the network layer (Section III-A), a max-flow min-variance algorithm is proposed to maximize network throughput and fairness in the heterogeneous duty-cycle network. In the MAC layer (Section III-B), with the time synchronization service from hardware, a localized TDMA MAC called TreeMAC [2] is propose to synchronizes nodes' duty-cycles in the "multi-path" flow network. A low-cost WWVB radio receiver is proposed to provide the accurate time synchronization. The complete hardware and software system has been implemented in both real sensor network testbed and simulators.

A. Network Algorithm for Throughput Maximization and Fairness

The software design involves both network and MAC layer. Subsection III-A1 and III-A2 describes our collaborative data delivery algorithm which maximizes network flow and fairness

on heterogeneous duty-cycled network. In subsection III-B, a localized TDMA MAC protocol is presented for duty-cycle synchronization to support the above collaborative data delivery protocol.

1) Loop-free Max-flow Algorithm for Throughput Maximization: In this section, we present our "loop-free" maximum flow algorithm to collaboratively utilize heterogeneous duty-cycles for throughput maximization. Besides, various optimizations are also given in Algorithm 1, including eliminating the flow loop to improve the energy efficiency.

The distributed "Push-Relabel" algorithm in [3] solves the max-flow problem under the edge-capacited network. Initially, the source node has a valid height and start a "push" operation. "Relabel" operation adapts the height in each node, so that data flow seeps through the network just as water flow through a terrain. However, it can not directly apply to our problem due to two main reasons. First, the capacity constraint is not on edges, but on nodes. In our problem, the constraint is that the total flow amount of the incident edges of a node u can not exceed its capacity. As stated earlier, each node has a known capacity constraint, while each edge's capacity is unknown and depends on its surrounding nodes. Second, there are multiple sources and single sink. All nodes in the network except the sink sense the environment and deliver data toward the sink, and they need to get a fair flow allocation.

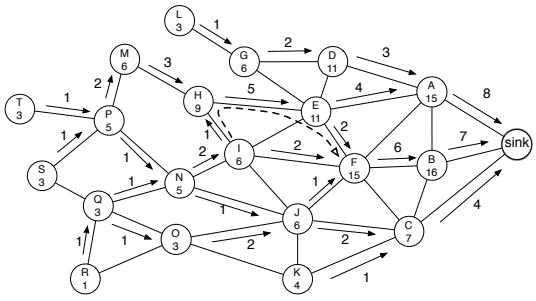


Fig. 4. Max-flow network with long push loop

Besides the major differences, it is also worthy to point out the "Long Push Loop" problem, which hinders the efficient data delivery. It is not hard to see that the "PUSH" operation in original push-relabel algorithm chooses a valid path in an arbitrary way, which may result in loops. And we have found the "Long Push Loop" problem in the arbitrary "PUSH" operation, illustrated in Figure 4. In Figure 4, node I with capacity 6 has 2 inflows. Assuming it has four admissible leaving edges in the residual graph, (I, H), (I, J), (I, E) and (I, F), node I can push its excessive flow to any of these edges. If node I arbitrarily selects node H to push, the flow may go through H, E and back to F. Compared to pushing flow directly to node F, this arbitrary push may result in longer data delivery path. And if each node maintains the arbitrary selection for PUSH method, the long-loop path may exist.

In Algorithm 1, we present our loop-free push-relabel max-flow algorithm. At the beginning, the height of every node is initialized as 0. And a source rate D_u is assigned to every node except sink. Notice that, this step is the same as adding a virtual source connecting every source node with link capacity D_u. After that the algorithm executes the push-relabel operation on each overflowed node to solve the max-flow algorithm. A key difference is that in line 4 of Algorithm 1, the $C_f(x, y)$ is determined by the minimum node residual capacity of node x and y: C_x and C_y. The reason is that one node may deplete its energy or reach its node capacity before the transmitting data rate exceeds the limit of its incident link. In the Algorithm 1, each node implements the PUSH operation with a careful selection on neighbors with residual capacity. The neighbors with the shortest path to sink and larger node degree will have higher priority. The rationale of this selection strategy is that the neighbor with the smallest hop-count and largest degree will likely yield more opportunities to deliver data to the sink. When the Algorithm 1 terminates, it generates a maximum flow assignment in the network, which can be easily converted to each individual node's maximum transceiving data rate (e.g., T_u, R_u) thereafter.

Algorithm 1 Loop-free Max-Flow Algorithm

procedure: *InitializeNetwork(V,E)*

1: **for** each vertex $x \in V \setminus sink$ **do**
2: add virtual link connecting x and virtual source
3: $excess[x] \leftarrow D_x$; $h[x] \leftarrow 0$
4: **end for**

procedure: *Push-Relabel(V,E)*

while $\exists$ x $\in$ V $\setminus$ $sink$ && $excess[x] > 0$ **do**
2: $U_x \leftarrow \emptyset$; $W_x \leftarrow N_x$
while $W_x \neq \emptyset$ **do**
4: **if** $C_f(x, y) > 0$ **then**
$U_x \leftarrow U_x \bigcup y$
6: **end if**
$W_x \leftarrow W_x$ - y
8: **end while**
now U_x is the set of node x's neighbors with residual capacity. Sort U_x in ascending order of their hop-count to sink and descending order of node degree (when hop-count is same).
10: update the shortest path from x to $sink$
if $U_x \neq \emptyset$ **then**
12: **while** $excess[x] > 0$ && $U_x \neq \emptyset$ **do**
y $\leftarrow U_x[head]$
14: **if** $h[x] = h[y] + 1$ **then**
$d_f(x, y) \leftarrow min\{excess[x], C_f(x, y)\}$
16: Push($d_f(x, y)$, x, y)
excess[x] $\leftarrow$ excess[x]-$d_f(x, y)$
18: **end if**
$U_x \leftarrow U_x$ - $U_x[head]$
20: **end while**
else
22: Relabel(x): h[x] $\leftarrow$ 1+ $min\{h[y] : (x, y) \in E_f\}$
end if
24: **end while**

We now first show that the loops in flow path are eliminated by our algorithm.

Lemma 1: In the flow network generated by Algorithm 1, each of its flow paths is loop-free.

Proof: The loop-free flow path can be proved by contradiction. Assuming there is a loop flow path $u, v, ..., w$, where the edge (u, w) also belongs to the original graph. The "PUSH" method given by Algorithm 1 can select v as the parent of u since the shortest path has highest priority. It implies that the flow path $u, v, ..., w, ..., sink$ is the shortest path. On the other hand, edge (u, w) is also selected due to the shortest path preference, which means these two shortest paths shall have equal length. We then assume that the length of path from w to the sink is l, the length of flow loop is the summation of l and length of $u, v, ..., w$. Contradiction is induced. ■

To prove Algorithm 1 does generate maximum network flow when it terminates, we need to show that it keeps the height function definition (which is a key condition in Golberg-Tarjan's push-relabel algorithm [4]).

Lemma 2: In Algorithm 1, the substitution of node capacity for link capacity still keeps function h as a height function.

Proof: In the step of $InitializeNetwork(V, E)$, the h is initialized as a height function. Therefore, the h will be valid if both $RELABEL$ and $PUSH$ methods leave h a valid height function. First, we look at the $RELABEL$ operation, which is executed in the overflowed node. If there is a residual edge (x,y) $\in E_f$ which leaves x, then after the relabel operation on node x, $h[x] \leq h[y] + 1$ follows. Since we use the node capacity, there are no residual edges entering to an overflowed node. Thus, the $RELABEL$ operation holds h as a valid height function. Second, $PUSH$ can only occur in the edge (x, y) with $h[x] = h[y] + 1$, which results in a residual link (y, x) with $h[y] = h[x] - 1 \leq h[x] + 1$. Thus, the height function still holds after the $PUSH$ operation. ■

Proven the height function retains as a valid function during Algorithm 1, we then prove the correctness of our algorithm. In other words, the maximum network flow can be achieved at the time when algorithm terminates.

Lemma 3: When Algorithm 1 terminates, the computed preflow in the network is a maximum network flow.

Proof: Detailed proof refers to technical report [6]. ■

Besides the correctness of algorithm, we also find the upper bound for algorithm's time complexity.

Lemma 4: (Upper bound of saturating pushes) Let $G = (V, E)$ be a flow network generated by Algorithm 1, the number of saturating pushes in Algorithm *Push-Relabel(V,E)* are less than $2|E|$.

Proof: Detailed proof refers to technical report [6]. ■

Lemma 5: (Upper bound of nonsaturating pushes) Let $G = (V, E)$ be a flow network generated by Algorithm 1, the number of nonsaturating pushes in Algorithm *Push-Relabel(V,E)* is bounded by $4|V|(|V|^2 + |E|)$.

Proof: Detailed proof refers to technical report [6]. ■

Consequently, we have the following theorem:

Theorem 6: The Algorithm 1 generates a loop-free maximum network flow and terminates within $O(|V|^2|E|)$ operations.

Proof: Detailed proof refers to technical report [6]. ■

In Algorithm 1, we describe it in a centralized way for simplicity of presentation and understanding. It can be easily implemented as a distributed network protocol, just like the original push-relabel algorithm [3] can be implemented in distributed fashion.

2) Flow Balance Algorithm for Fairness: Though Algorithm 1 maximizes network throughput in a duty-cycled network, it does not guarantee flow fairness. In Algorithm 2, we propose a flow balance algorithm, providing fair flow assignment among nodes while maintaining maximum network flow. Unlike some other existing works which rely on the predetermined tree structure, our min-variance flow balance algorithm can apply to general data collection structure.

Algorithm 2 Distributed Min-Variance Flow Balance

procedure: *MinVariance(u)*

while $(f_{min} \neq f_{max})$ *and* $\Delta f_{min} > 0$ **do**
  $F_u \leftarrow \{f_{min}, f_{max}, \Delta f_{min}\}$
  $F_u^{'} \leftarrow \{f_1, f_2, ..., f_k\}$
  $f_u^{max} \leftarrow f_{max}$; $f_u^{min} \leftarrow f_{min}$
  Δf = *FSAP*(f_u^{min},f_u^{max})
  if Δ = max $\{\Delta f, \Delta f_{min}\} > 0$ **then**
    $f_u^{min} \leftarrow f_u^{min} + \Delta$; $f_u^{max} \leftarrow f_u^{max} - \Delta$
    $Update$ Δf_{min} along the path
  else
    $Find$ new f_{min} by BFS and update Δf_{min}
  end if
end while

procedure: *FSAP*($f_x^{min}(*,*)$,$f_y^{max}(*,*)$)

$U^{'} \leftarrow x$; $P \leftarrow \emptyset$; $\Delta f \leftarrow 0$
while $\delta_f(x, y) < |V|$ **do**
  if G_f contains an admissible edge $(U^{'}, U^{''}) \in E_f$ **then**
    let $(U^{'}, U^{''})$ be an admissible edge in E_f
    $\pi(U^{''}) \leftarrow U^{'}$; $U^{'} \leftarrow U^{''}$
  end if
  if $U^{'} = y$ **then**
    use π to find an augmenting path P
    $\Delta f \leftarrow \min((f_y^{max}(*,*) - f_x^{min}(*,*))/2, C_f(P))$
    return Δf
  else
    Relabel($U^{'}$) ; $U^{'} \leftarrow \pi(U^{'})$
  end if
end while
return Δf

In order to apply the distributed flow balance algorithm, each node maintains two flow vector: $F_u(f_{min}, f_{max}, \Delta f)$,

which is the accumulated min flow, max flow and the adjustable flow amount; $F_u^{'}(f_1, f_2, ..., f_k)$, which are the flows from its direct neighbors. The overall idea of distributed Algorithm 2 is to iteratively select the f_u^{max} and f_u^{min} flows out of flow set, and try to find the shortest augmenting path between them. In each iteration of $MinVariance$ of Algorithm 2, the routine $FSAP(f_x^{min}, f_x^{max}, G)$ will find a shortest augmenting path with either positive or zero adjustable flow amount. If an augmenting path is found in $FSAP()$ method, the $FSAP()$ returns a positive Δ_f. Then these two flows can be balanced by letting f_u^{max} yield Δ_f amount of flows to f_u^{min}. This is described in line 6 to 8 of procedure: $MinVariance$. After the first pair of f_u^{max} and f_u^{min} has been solved, we rearrange the flow set accordingly, and repeat the procedure until there is no more adjustable flows in the network. Otherwise, we will update the f_{min} in line 10 by removing the unadjustable f_{min} from corresponding node, i.e. node x, and recompute the f_{min} for node x based on its $F^{'}$.

The time complexity for $FSAP()$ is $O(|E|)$, and it have at most $|V|$ iterations. And the algorithm will terminate when f_{min} equals to the f_{max}, which is bounded by size of vertex set $|V|$. Therefore, the complexity of Min-Variance Flow Balance Algorithm is $O(|V|^2|E|)$.

B. Localized TDMA MAC Protocol for Duty-cycle Synchronization

To actually achieve the performance of aforementioned algorithm, duty-cycle need to be synchronized in MAC layer. We proposed a localized TDMA MAC protocol with the objective of synchronizing duty-cycles on the selected network flow paths and eliminating media access contentions. In order to support the time synchronization in TDMA MAC protocol, we design a low-power hardware by utilizing the WWVB atomic clock receiver. We select the CME 6005 [7] as the receiver chip, which is of only about 10 dollars, in our hardware design. Its current consumption is less than $100\mu A$ in full active mode with the voltage supply of 3.0V, which is much lower energy cost than software-based time synchronization protocols, like FTSP [8]. The software-based time synchronizations need to turn radio on and exchange time messages periodically so that clock drift of sensor nodes can be compensated.

The MAC protocol is inspired from TreeMAC [2] - a localized TDMA MAC protocol. The main idea is to utilize parent-child relationship in a data collection tree to minimize the signaling overhead. A time *cycle* is divided into *frames* and frame into *slots*. The parent determines each children's *frame* assignment based on their relative bandwidth demand, and each node calculates its own *slot* assignment based on its hop-count to the sink. The control message is between parent and child only. This simple yet effective 2-dimensional frame-slot assignment algorithm can avoid schedule conflicts in network flow of shortest path tree.

However, TreeMAC [2] can not directly apply to the "multi-path" flow network, as flow network topology is not a tree structure. Fortunately, such a many-to-one flow network graph can be virtualized as a tree structure. Basically, a node u with k parent can be virtualized as k virtual nodes $\{u_1, u_2, \cdots, u_k\}$, and then each u_i ($1 \leq i \leq k$) now have a single parent, as illustrated in the figure 5. This virtualization can be further extended: for every node u in the network (even if it has a single parent or single path to the sink), assuming it has m-unit flows to forward, it can also be virtualized as m virtual nodes $\{u_1, u_2, \cdots, u_m\}$. Then each link with m flow units can be virtualized accordingly and a virtual tree structure emerges.

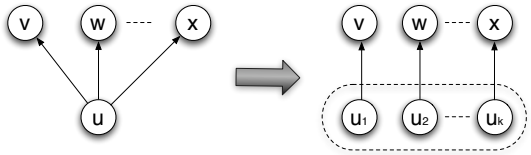


Fig. 5. Virtualization example.

The frame-slot assignment algorithm of TreeMAC can apply to the virtual tree structure now. For slot assignment, we can directly apply the slot assignment algorithm mentioned above. For frame assignment, assume each flow has a flow ID f_{ID}, we can assign all nodes on the flow path with the f_{ID}-th frame. Now, different flows are guaranteed to be conflict-free, since they have different frames per cycle. However, this will require that the time cycle size is bigger than the number of flows. This is not a problem, because the number of flows is constrained by the sink node's capacity.

Another concern on the difference between tree and virtual tree is: in a tree structure, every node only has one parent to forward data, and its hop-count can be determined unambiguously; but in a flow network, a node u may have a different hop-count through different selected flow paths: $(u, v, ..., y, sink)$ and $(u, w, ..., y, sink)$. Thus the virtual nodes u_i and u_j from the same node u may have a different hop-count too. Fortunately, this is also not a problem. This just means that the original node u will get different slots in different frames, instead of the same slot in different frames.

Theorem 7: The duty-cycle synchronized MAC protocol based on TreeMAC ensures conflict-free transceiving schedules in maximum flow networks.

Proof: Using TreeMAC frame-slot assignment in the virtual tree, the frames assigned to different virtual nodes are non-overlapping, if those nodes do not have ancestor-descent relationship. Thus, there must exist no schedule conflict between different unit-flow paths. In addition, according to Lemma 1, our flow paths have no loops. Since parent and child have no schedule conflicts in TreeMAC, any two nodes in the same flow path also do not have schedule conflicts. ■

IV. System Evaluations

A. Experimental Evaluations

In this section, we evaluate our system design and implementation, including hardware design, MaxFairFlow algorithm (Loop-free Max-flow and Min-variance fairness algorithm) and localized TDMA, by comparing to the commonly used

TinyOS protocol stack in three evaluation criteria: *Data Throughput*, *Data Delivery Latency* and *Energy Efficiency*.

We show our experimental testbed in Figure 6, topology in Figure 7. The WWVB antenna is mounted at the top of the pole, and TelosW [1] is attached to the WWVB receiver circuit board. We implement and deploy our energy synchronized system in the sensor network testbed of 15 TelosW nodes, which are placed in a square 10*10 meters area with multihop topology. The sink node is placed in the upper left corner.

Fig. 6. Outdoor experiment test-bed of 15 nodes synchronized by WWVB radio.

In Figure 7, each node is indicated by a tuple of ID and node capacity. Low radio transmission power is configured to form a multihop duty-cycled network, with maximum 5 hops.

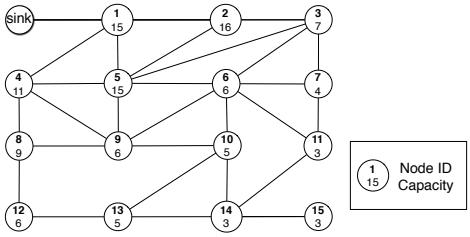


Fig. 7. The snapshot of topology of testbed. The upper number inside the circle denotes the node ID, the lower one is the node capacity.

1) Data Throughput: We analyze the data throughput to validate the data collection efficiency of our protocol. Figure 8(a) shows the comparison of the data throughput between the CTP-XMAC and MaxFairFlow-TDMA algorithm. With the total sending packet number of 3600 in each node, the minimum node throughput for CTP-XMAC is $2,916$, which is approximately 81.0% in terms of packet delivery ratio. It happens on node 13 with hop-count 4. And the maximum throughput is $3,519$ on node 1. In the network, the total throughput is 6.6% lower than that of MaxFairFlow-TDMA.

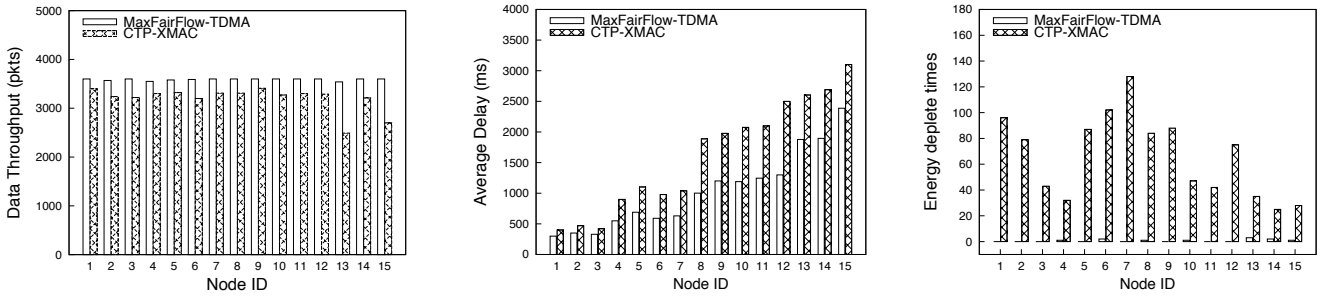
The reason for the higher packet delivery ratio is that MaxFairFlow-TDMA algorithm takes advantages of the "multi-path" for collaborative data forwarding. The max-flow and min-variance algorithm can achieve the optimal forwarding paths for each node, bypassing the congested node. For each node, our algorithm can increase the forwarding probability of each packet, while for the entire network, the data traffic is balanced over different delivery paths.

2) Data Delivery Latency: The low-duty-cycled network poses a critical challenge on the timeliness of delivered data. We have evaluated the data delivery latency in the network against multiple hopcounts. Figure 8(b) shows that the latency of CTP-XMAC increases proportionally to the hopcount, while the rate of increasing latency in our algorithm is much slower. It also shows that the maximum latency for CTP-XMAC is $3100\ ms$, which is 5 hops away from sink. But the maximum for our MaxFairFlow-TDMA protocol is only $2387\ ms$ at node 15. It is because that in MaxFairFlow-TDMA, nodes wake up in the designated slots with synchronized time, without any delay between the wake-up time of sender and receiver. Overall, data delivery in TDMA with assigned synchronized slots can provides better latency for QoS required applications.

3) Energy Efficiency: The energy efficiency is evaluated in this section. Instead of estimating energy from communication packet number or attaching the oscilloscope physically to the mote, we exploit the on-board energy meter to measure the real-time energy consumption rate [1]. The benefit is that it evaluates all the real energy consumption including RF transceivers and WWVB radio.

Figure 8(c) shows the total number of times nodes deplete their quota energy within experiment. Respecting the planned lifetime, each node has a derived energy cap in each time frame. With the realtime energy consumption measurement from energy meter, a node can decide if it runs out of the derived "quota energy". In Figure 8(c), the maximum energy deplete times for CTP-XMAC approach is 128 times, and every node has, at least 30 times, depleted its own energy. The Figure 8(c) shows that the proposed algorithm almost never deplete the quota energy, less than 5 times in node $6, 13$ and 14 respectively. It is clear that different duty cycles and asynchronous wake-up times give the asynchronous scheduling protocol (e.g. CTP-XMAC) a challenge, which can cost more energy in holding on the radio and waiting for the receiver to wake up. The reasons for the overall advantage of energy consumption gained in MaxFairFlow-TDMA are two-folded: (i) wake-up schedules are synchronized among different duty-cycled nodes; (ii) energy consumption is more balanced through collaborative multi-path selection in the Max-Flow Algorithm in section III-A1 and Min-Variance Flow Balance Algorithm in section III-A2.

Moreover, we examine the network lifetime under different protocols. Network lifetime is the network up-time from boot-up to the time when the first node dies. In Figure 9(a), the first node goes down after 180 hours in CTP-XMAC, and the maximum lifetime for nodes in the network is 221 hours. In MaxFairFlow-TDMA, nodes were running 231 hours before the first node depletes its residual energy, which is even longer than the maximum node uptime in CTP-XMAC. Extended network lifetime is due to that our MaxFairFlow-TDMA algorithm balances the network traffic through multiple paths, and deliver the sensing data in a collaborative way. In Flow Balance Algorithm of section III-A2, the assigned



(a) Data throughput. (b) Average delivery delay in the network. (c) The number of times of depleting quota energy.

Fig. 8. Experimental evaluation of data delivery ratio (a), latency (b) and nodes energy consumption(c).

flows are balanced in each flow path, achieving maxmin and minmax fairness. Consequently, the algorithm balances the energy loads in multiple paths as well. All of these contribute to improving the network lifetime.

B. Simulation Evaluations

To show the scalability of our proposed algorithm, we have evaluated the performance on a larger scale with TOSSIM. All the sensor nodes are placed in a 100*100 square grid, with sink node on the left upper corner.

1) Network Delay: The average and maximum packet delay is evaluated with network size ranging from 20 to 180. And the maximum delay is affected by the maximum hopcount in the network. This is why the maximum delay increases from 3100 ms of 20-node network to 20000 ms in a 180-node network in Figure 9(b). And the delay of MaxFairFlow-TDMA scheduling has a much slower increasing slope, ending up with the 12000 ms delay in the network with 180 nodes.

In terms of average delay over the different network sizes, both of the two protocols have the slow increasing trend. This is because the average hopcount does not increase sharply as the network size goes larger. It is important to mention that the MaxFairFlow-TDMA method still gets an advantage of 25.2% in the average packet delay. That is because that collaborative multi-path selection makes the distribution of nodes more balanced, resulting in a smaller average hop count.

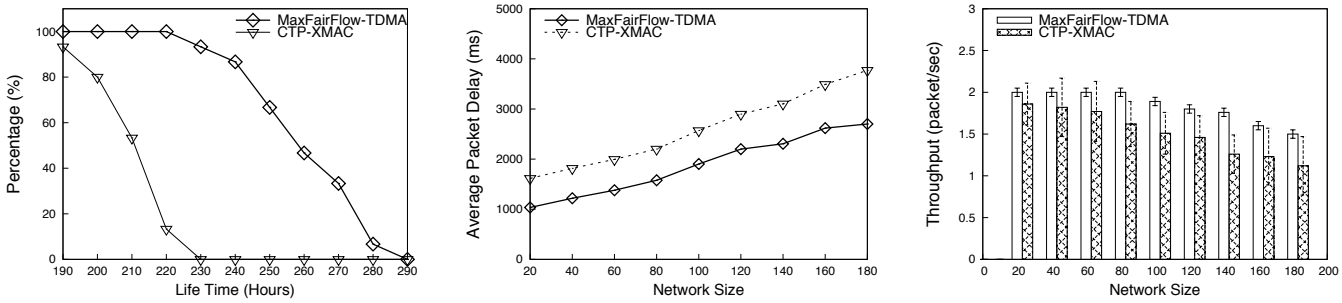
2) Network Throughput: We have measured the average throughput per node in different network sizes. In Figure 9(c), the bar denotes the average throughput for each node of a given network size, while the error reflects node throughput deviation in that network size. As the network size scales, in CTP-XMAC, it is observed that the throughput per node starts to decrease at the size of 40, and continue declining to the 1.12 packets / second in each node, at the network size of 180. For the MaxFairFlow-TDMA, the network starts to saturate at the size of 80, and the throughput for each node decreases until 1.5 packets per second. Moreover, it is observed that the variance of throughput for each node remains within the value of 0.05 packets per second in MaxFairFlow-TDMA. However, the variance for CTP-XMAC is much higher, in which the maximum variance is 0.36 packets per second, with average variance of 0.30. It is because the Flow Balance Algorithm of section III-A2 can efficiently even out the traffic flows and take advantages of multi-path for delivering data packets.

V. Related Work

The maxmin optimal rate assignment for network fairness have been studied in a few related works. In [9], it presents an iterative linear programming solution, which has the same goal as our max-flow and min-variance algorithm. They proved that there is one and only one such assignment for a given configuration of the sensor networks. Its worst-case time complexity is $O(|V|^2C(|E|))$, where $C(|E|)$ is the complexity of linear programming with $O(|E|)$ variables and constraints. The complexity of $C(|E|)$ is not given in the paper, however, it is known to be polynomial $O(|E|^\alpha)(\alpha \geq 1)$, which is at least $O(|E|)$. Recall that the time complexity of our max-flow and min-variance algorithm is bounded by $O(|V|^2|E|)$ which is lower. [9] did not provide any simulation or test bed results. In [10], it aims at designing a solution for fair and high throughput data extraction from all nodes in the presence of renewable energy sources. It gives both a centralized algorithm and distributed algorithm. The centralized algorithm is similar to [9], except that they assume resources are dynamic and vary over the time. However, the asynchronous distributed algorithm assumes that the data collection flow is a tree structure, so it may not achieve maximum throughput as discussed earlier in the paper.

Some related works have studied congestion control and fair rate assignment schemes. [11] proposes a distributed rate allocation scheme (IFRC) without end-to-end reliability. IFRC scheme detects the congestion point, and advertises the congestion by overhearing each other's messages, finally converges to a rate assignment, which is fair and avoids congestion collapse. A more recent work [12] presents a reliable centralized algorithm (RCRT) to allocate the fair date rate to each node, which can achieve more than twice the rate achieved by IFRC. Those works are not optimized for heterogeneous duty-cycled networks, where optimization in MAC and routing layer is necessary.

The related works on MAC protocol design for duty-cycled sensor networks are plentiful. A common goal is to reduce the energy consumption, while increasing the possibilities of rendezvous between sender and receiver. [13] provides a



(a) The CDF of expected life time in the network. (b) Maximum and Average Delay in different network sizes. (c) Average throughput per node against different network sizes.

Fig. 9. Expected life time (a) and simulation results on different scale of sensor networks (b) & (c).

distributed transmission slot scheduling protocol to save power consumption in wireless sensor networks. But the control message would be appreciable under large sensor network, and "hidden terminal" could cause the collision of advertisement or confirmation messages. In B-MAC [14] and X-MAC [15], preambles are used by sender to wake up receiver. In contrast, RI-MAC [16] let the receiver wake up the sender to transmit and can avoid buffer overflow and data losses. The current consumption of preambles or control messages is not necessarily small. Our proposed algorithm differs in synchronizing the wake-up scheduling by allocating wake-up time slot based on MaxFair flow ID and nodes' hopcounts, reducing the energy consumption on wake-up scheduling.

[17] designs a localized energy synchronization control mechanism that shuffles or adjusts the working schedule of a node to optimize delays in the presence of varying energy budget over time. In their work, a delay model for cross-traffic at individual nodes is derived and a stair effect is observed in low-duty-cycle networks. More recently, [18] proposes a proactive solutions for minimizing sink-to-node communication delay in energy-harvesting sensor networks. They introduce a delay maintenance algorithm with minimum energy consumption. In both works, they assume node duty-cycles can be augmented to reduce delays. We note that in an energy-synchronized network, each node's duty-cycle shall be determined by its residual energy (or energy harvesting rate) and augmenting duty-cycles may sacrifice lifetime. Therefore, we utilizes multiple paths to minimize delays.

VI. Conclusions

In this paper, we presented a comprehensive system design involving both network/MAC layer and hardware design for collaborative data delivery in energy-synchronized sensor networks. The proposed max-flow min-variance algorithm achieves maximum network flow with fair rate assignment and can be applied periodically if the node capacity is changing over the time. The evaluations have shown the advantages of this design, which not only increases network delivery ratio and fairness, but also increases network lifetime, compared to existing protocols. Moreover, this work also provides an estimation of network throughput and fairness under fixed duty-cycles or lifetime constraints.

References

[1] G. Lu, D. De, M. Xu, W.-Z. Song, and B. Shirazi, "TelosW: Enabling Ultra-Low Power Wake-On Sensor Network," in *INSS 2010*, Jun. 2010.

[2] W.-Z. Song, R. Huang, B. Shirazi, and R. Lahusen, "TreeMAC: Localized tdma mac protocol for high-throughput and fairness in sensor networks," in *The 7th Annual IEEE International Conference on Pervasive Computing and Communications (PerCom)*, Mar. 2009.

[3] T. L. Pham, I. Lavallee, M. Bui, and S. H. Do, "A Distributed Algorithm for the Maximum Flow Problem," in *Proceedings of the 4th International Symposium on Parallel and Distributed Computing*, Jul. 2005.

[4] A. V. Goldberg and R. E. Tarjan, "A new approach to the maximum flow problem," *Journal of the ACM (JACM)*, vol. 35, pp. 921–940, Oct. 1988.

[5] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein, *Introduction to Algorithms*. McGraw-Hill Book Company, 2001.

[6] M. Xu, W.-Z. Song, and G. Xing, "Collaborative Data Delivery in Energy-Synchronized Sensor Networks," available at http://sensorweb.cs.gsu.edu/∼xum/pub/EnergySynchronized-TR.pdf, Tech. Rep., Dec. 2010.

[7] CME6005:http://www.c-max-time.com/products/showProduct.php?id=2.

[8] M. Maróti, B. Kusy, G. Simon, and A. Lédeczi, "The flooding time synchronization protocol," in *Proc. 2nd ACM conference on Embedded networked sensor systems (SenSys)*, Nov. 2004.

[9] S. Chen, Y. Fang, and Y. Xia, "Lexicographic Maxmin Fairness for Data Collection in Wireless Sensor Networks," *IEEE TRANSACTIONS ON MOBILE COMPUTING*, vol. 7, pp. 762–776, Jul. 2007.

[10] K.-W. Fan, Z. Zheng, and P. Sinha, "Steady and Fair Rate Allocation for Rechargeable Sensors in Perpetual Sensor Networks," in *SenSys 2008*, Raleigh, North Carolina, USA, Nov. 2008.

[11] S. Rangwala, R. Gummadi, R. Govindan, and K. Psounis, "Interference Aware Fair Rate Control in Wireless Sensor Networks," in *SIGCOMM*, Sep. 2006.

[12] J. Paek and R. Govindan, "RCRT: Rate-Controlled Reliable Transport for Wireless Sensor Networks," in *SenSys*, Nov. 2007.

[13] B. Hohlt, L. Doherty, and E. Brewer, "Flexible power scheduling for sensor networks," in *Proceedings of the third international symposium on Information processing in sensor networks (IPSN)*, 2004.

[14] J. Polastre, J. Hill, and D. Culler, "Versatile Low Power Media Access for Wireless Sensor Networks," in *The 2nd ACM Conference on Embedded Networked Sensor Systems (SenSys)*, 2004.

[15] M. Buettner, G. Yee, E. Anderson, and R. Han, "X-MAC: A Short Preamble MAC Protocol For Duty-Cycled Wireless Sensor Networks," in *SenSys 2006*, Boulder, Colorado, USA, Nov. 2006.

[16] Y. Sun, O. Gurewitz, and D. B. Johnson, "RI-MAC: A Receiver Initiated Asynchronous Duty Cycle MAC Protocol for Dynamic Traffic Load." in *Proc. 6th ACM conference on Embedded networked sensor systems (SenSys)*, Nov. 2008.

[17] Y. Gu, T. Zhu, and T. He, "ESC: Energy Synchronized Communication in Sustainable Sensor Networks," in *The 17th International Conference on Network Protocols*, Oct. 2009.

[18] Y. Gu and T. He, "Bounding Communication Delay in Energy Harvesting Sensor Networks," in *ICDCS*, Jun. 2010.