Networking Wireless Energy in Embedded Networks

YASRA CHANDIO, Department of Computer Science, LUMS
JÓ ÁGILA BITSCH, RWTH Aachen University
AFFAN A. SYED, INNEXIV and FAST National University
MUHAMMAD HAMAD ALIZAI, Department of Computer Science, LUMS

9

Wireless energy transfer has recently emerged as a promising alternative to realize the vision of perpetual embedded sensing. However, this technology transforms the notion of energy from merely a node's local commodity to, similarly to data, a deployment-wide shareable resource. The challenges of managing a shareable energy resource are much more complicated and radically different from the research of the past decade: Besides energy-efficient operation of individual devices, we also need to optimize networkwide energy distribution. To counteract these challenges, we propose an *energy stack*, a layered software model for energy management in future transiently powered embedded networks. An initial specification of the energy stack, which is based on the historically successful layered approach for data networking, consists of three layers: (i) the *transfer* layer, which deals with the physical transfer of energy; (ii) the *scheduling* layer, which optimizes energy distribution over a single hop; and (iii) the *network* layer, creates a global view of the energy in the network for optimizing its networkwide distribution. As a contribution, we define the interfacing APIs between these layers, delineate their responsibilities, identify corresponding challenges, and provide a first implementation of the energy stack. Our evaluation, using both experimental deployments and high-level simulations, establishes the feasibility of a layered solution to energy management under transient power.

CCS Concepts: • **Computer systems organization** → **Embedded systems**; • **Networks** → Network reliability;

Additional Key Words and Phrases: Energy plane, energy networking, energy stack

ACM Reference format:
Yasra Chandio, Jó Ágila Bitsch, Affan A. Syed, and Muhammad Hamad Alizai. 2018. Networking Wireless Energy in Embedded Networks. *ACM Trans. Sen. Netw.* 14, 2, Article 9 (April 2018), 31 pages.
https://doi.org/10.1145/3185753

1 INTRODUCTION

Will the vision of a pervasive and perpetual sensing infrastructure be truly possible while we keep energy management as a necessary but carelessly intertwined part of the network stack? Energy has always been a constraining resource influencing both research and industrial implementations of wireless sensor networks (WSN) and the Internet of Things (IoT) space. This has resulted in energy awareness seeping into the networking stack. These stacks treat data transport as a first-class

Authors' addresses: Y. Chandio, Department of Computer Science, LUMS, DHA, Lahore, Punjab, 54792, Pakistan; email: yasra.chandio@lums.edu.pk; J. Á. Bitsch, COMSYS, RWTH Aachen University, Ahornstr. 55, 52074, Aachen, Germany; email: jo.bitsch@cs.rwth-aachen.de; A. A. Syed, Fast National University, A.K. Brohi Road, H-11/4, Islamabad, Pakistan; email: affan.syed@innexiv.com; M. H. Alizai (corresponding author), Department of Computer Science, LUMS, DHA, Lahore, Punjab, 54792, Pakistan; email: hamad.alizai@lums.edu.pk.

https://doi.org/10.1145/3185753

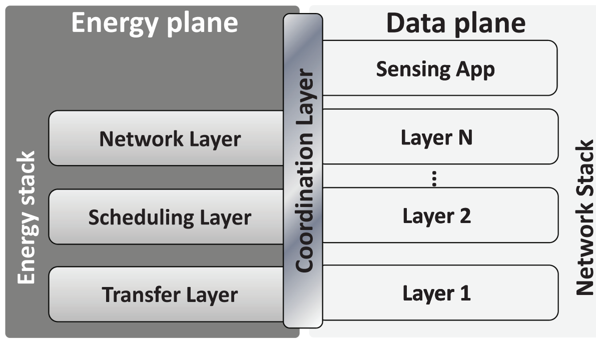


Fig. 1. Logical decoupling of energy and data planes: The *energy stack* is responsible for physical transfer of energy (*transfer layer*), optimizing energy distribution over a single hop (*scheduling layer*), and for creating a global energy view to optimize network wide distribution (*network layer*).

objective; consequently, the energy constraints show up intertwined as an afterthought with no clear architectural separation. While energy was a local commodity, tightly coupled to a battery or harvesting source, this approach did not preclude building energy-aware network protocols.

However, with the realization of wireless energy transfer (WET) and its use-cases in embedded networks [19, 34], energy now becomes a deployment-wide shareable resource. We can now physically "decouple" an energy source (e.g., a laser transmitter) from where it is consumed (e.g., an in-situ sensing device), allowing a heterogenous master/leaf network architecture for distributing energy [6]. This separation of functionality for data and energy transport allows us to start reasoning about a logically decoupled *energy plane* (energy sources and distribution) to provide a clean separation of functionalities. A decoupled energy plane will allow application writers to focus on delivering *data plane* functionality (the spatio-temporal sensing, computation, and communication requirements of an application), while relying on a reliable interface; an energy-data *coordination* application programming interface (API). Moreover, we can now best design a software stack primarily written to incorporate the concerns, constraints, and capabilities intrinsic to WET-influenced energy management including energy transport across a battery-free network deployment.

In this article, we thus articulate the case for an *energy stack* to formally implement the data/energy plane separation; since this is a radical new path, we tread it by focusing on a *tiered* network architecture (cf. Figure 2) for WET enabled deployments. The energy and data stacks interact with each other through a vertically integrated *coordination layer* (Figure 1), which is responsible for bi-directional data and control information to be exchanged in a seamless manner. We build the energy stack using a layered approach, inspired by the successful history of the layered software architecture for data networking operations. The three layers—*transfer*, *scheduling*, and *network*—provide the basis for building and using an energy stack for a WSN/IoT device. We evaluate an instance of this stack for a specific WET technology (as an example transfer layer), using empirical micro-experiments to guide simulated macro-experiments regarding the different possible implementations of the scheduling layer. While we do not fully evaluate multiple network layer protocols, we explore some simple ideas indicating the richness of opportunity available in optimizing network operation (in the data plane) by creating a global energy view of the network.

The contributions of this article thus are as follows:

- The proposal of a layered energy stack for the energy plane, an API between layers of the energy stack, as well as an API for interactions between energy and data planes through a vertically integrated coordination layer (Section 3).

- Building and evaluating the *first* decoupled WSN software stack to bring home the realization of this new energy stack. To this end we propose, implement, and evaluate the following:
  (1) An instance of the transfer layer, achieving 7.4% duty cycle and supporting up to 40 nodes for a single 0.8W WET source (Section 4).
  (2) Three scheduling layer energy distribution algorithms under different application scenarios. We observe that, while no single algorithm is best for all the evaluated application scenarios, a heuristic demand-supply fairness algorithm turns out to be better than a statistically rigorous algorithm in the most common WSN topology (Section 5).
  (3) A network layer algorithm demonstrating the benefit of having a global energy view. We show how leaf set assignments to masters can be changed dynamically to accommodate varying workloads and node churn while maximizing performance (Section 6).

We discuss related work in Section 7 before concluding our article in Section 8. We deliberately target a broader view of the energy stack, its layers, and the challenges therein, rather than in-depth exploration of a particular layer. This is so as the true benefit of having a layered architecture will arise not from the algorithm we propose here, but the utilization of this separation to build better and more robust embedded networked systems. Hence, our evaluation focuses on system level aspects instead of benchmarking a particular protocol or hardware parameter (like the optimal capacitor size), where we rely on existing literature for our selection.

2 BACKGROUND: THE NEED FOR REDEFINING ENERGY MANAGEMENT

Tiny and easily deployable computing devices in our physical environment are at the foundation of the vision of IoT, bringing billions of new wireless, embedded hosts to the Internet. To realize this hyper-connected IoT world, *we need to eliminate wires both for communication and energy transfer* as proposed by Nikola Tesla [33]. While communication networks are becoming increasingly wireless, eliminating the wires for energy transfer has not been as successful. Replacing wires with batteries that accumulate only a finite energy budget is strongly restricted by the form factor of an IoT device. Moreover, batteries do not enable perpetual operation, as desired in IoT, and replacing them is unfeasible (with remote deployments [7]) or impractical and dangerous (for biomedical implants [38]). These limitations of battery operated devices, mainly encountered during WSN[1] and IoT research in the past decade, rejuvenated research on energy harvesting and wireless energy transfer (WET). We are now at a stage where these technologies can cater to the energy budget needs of these small embedded nodes [4, 19].

2.1 Energy Harvesting vs. Wireless Energy Transfer

Energy harvesting is the conversion of ambient energy into electric energy. The availability of ambient energy in a given environment can be either due to the presence of natural sources, such as sunlight or vibrations, or due to intentional provisioning through WET using, for example, a radio transmitter. Energizing embedded devices solely through natural energy harvesting, while in principle allowing perpetually running networks of embedded systems, has its limitations: The availability of natural sources of harvestable energy is a function of the deployment environment. Depending on its characteristics, suitable sources of natural energy may simply not be at our disposal. Application scenarios may further constrain possible node locations, therefore rendering the benefits of energy harvesting moot, as they may not outweigh the additional complexity and costs. As an example, potential sources of natural energy are lacking in environments such as

[1] We categorize WSN as the most practical subclass of IoT throughout this article.

road tunnels [10], north-facing ridges [3], and deep green forests [30], where the applications of embedded network deployments are well established. Moreover, the nodes cannot be re-arranged in space to favor energy harvesting, as their locations are dictated by application requirements. These factors limit the range of applications where harvesting can be applied to extract energy from natural sources.

These issues mainly are caused by the tight coupling between the location of energy harvesting and the location where the sensing needs to occur. WET—that is, the ability to move energy wirelessly across space—can break such coupling, allowing network designers to exploit abundant energy sources available at places other than the locations of sensing. In addition, WET may distribute the available energy from energy-rich locations to energy-poor ones, counteracting an overall energy imbalance. Due to these properties, WET has emerged as a promising alternative to energize battery-free IoTs (i) to function without an intrusive power infrastructure and (ii) to enable potentially perpetual deployments. Recently, several technologies emerged that enable WET in WSN applications as well, such as laser [6], power LEDs [23], or radio transmissions [9, 19, 27].

2.2 The Need to Rethink Energy Management for Wireless Energy Transfer

While the main focus of existing research in this domain is on exploring the feasibility of a particular WET technology, we also need to rethink energy management as WET redefines the role of energy in the network. Energy, previously considered as a node's local commodity to be managed against sensing, computation, and communication needs, now becomes a deployment-wide shareable resource. Using WET, network designers can intentionally transport energy to points of interests where the available natural energy is scarce. Moreover, as opposed to natural energy harvesting, spatio-temporal availability of energy in WET enabled deployments becomes more deterministic, as it provides better control over how to distribute energy across a network.

Whereas the traditional harvesting of natural energy sources, which only requires extensions in the form of a harvesting unit impacting individual nodes, WET gives rise to significant network level challenges as well. Network designers have to decide to whom to transfer energy, in what amount of it, and using what technology; this set of questions is already well established in the networking domain with respect to data communication. Hence, apart from the communication network, we now also have to manage an "energy network" among the nodes. However, the requirements and challenges of such an energy network are completely different and thus cannot be directly mapped on to the communication stack, as we discuss below while elaborating on our proposal to develop a separate stack for networking energy.

3 DESIGN: ENERGY PLANE WITH AN ENERGY STACK

The conceptual leap influenced by the emergence of WET renders existing energy management techniques obsolete, requiring a thorough update to contemplate some elementary properties of the WET-based embedded network paradigm. For example, besides the energy efficient operation of individual nodes, we also need to strategically distribute energy in the network. To this end, our proposed design first logically decouples the energy and data planes, reflecting the physical decoupling of energy and data transport in the system software as well. This logical decoupling allows for efficient integration of WET in WSNs by co-designing and co-optimizing both the planes through a vertical coordination layer API. Hence, energy is no longer an interspersed data plane constraint but a separated service that can evolve and be managed independently to serve the data plane more efficiently. We then design a new software model for the energy plane with the aim to develop an energy management mechanism that can efficiently handle the *shareable* nature of energy in WET enabled deployments.

3.1 Software Model

While data plane management through the network stack and operating system (OS) is well established, we propose an *energy stack*—the key contribution of this work—that follows a well-defined layered approach for managing the energy plane in the decoupled system model. Our proposed solution, as shown in Figure 1, borrows heavily from the historically successful, layered model for data networking. The energy stack is responsible for networking energy, i.e., collecting, sharing, and serving energy distribution requests among nodes. The use of a layered approach brings about the benefits of traditional data networking to *energy networking*, such as reduced per-layer complexity, standardized interfaces, modular engineering, potential vendor interoperability, and accelerated evolution of this young but revolutionary technology. We initially propose a minimal energy stack comprising of three layers: a *transfer*, a *scheduling*, and *network* layer. This three-layered stack is sufficient to apply and demonstrate the feasibility of the layered concept for energy management. However, in the future, we do not leave out the possibility of reassembling the layers, their interfaces, and readjusting responsibilities as the technology evolves.

- The **transfer layer** is responsible for the physical transfer of energy between a transmitter and a receiver node. It controls hardware aspects, such as the transmission power and the modality of the energy transfer.
- The **scheduling layer** decides to whom, how much, and when energy shall be transferred over a single hop based on a given high-level, application-specific energy scheduling policy.
- The **network layer**, similarly to data networking, is responsible for maintaining an overall energy view of the network for global optimizations and efficiently serving energy requests in the network, potentially over multiple hops.

In addition, a vertical **coordination layer** acts as a bidirectional interface between the data and energy planes. Using this layer, the energy stack can, for example, communicate with the corresponding layers on other nodes using data transmission capabilities of the data plane. As an example, such an interaction at the network layer could include information exchange for creating a global energy view of the network. Similarly, in the opposite direction, an energy-aware application in the data stack can enquire about the state of energy buffer to fine-tune its operation.

Before discussing the API for this layered architecture, we first present the network model as it will facilitate a better understanding of the interfacing API.

3.2 Network Model

Due to the the limitations of the current energy harvesting and wireless transfer technologies, we redefine the roles of different nodes (see Figure 2). A homogeneous network model, which considers every node to be equally capable of energy harvesting and wireless transfer, is hardly feasible. For example, deployment constraints may create significant imbalances in the ability to harvest energy. As a result, a tiered network model is probably more appropriate, similarly to RFID systems. More capable nodes will be responsible for generating energy, perhaps from a renewable ambient source, and for transferring it to less capable in-situ devices that harvest and consume it.

We thus propose a logically two-tiered network model akin to a wide range of use-cases of WET-based practical sensor deployments [6, 19, 34]. In our network model, *tier-1* includes highly capable nodes (*masters*) responsible for generating energy, perhaps from a renewable ambient source or using mains power, and for transferring this energy to *leaves* in *tier-2*. We assume that nodes at tier-1 are at least capable of (i) harvesting (or generating) and (ii) transmitting energy and, in the case of directional energy transfer, (iii) reorienting their transmitters to energize multiple nodes. A future possibility could include equipping masters with mobility support to extend their transfer range

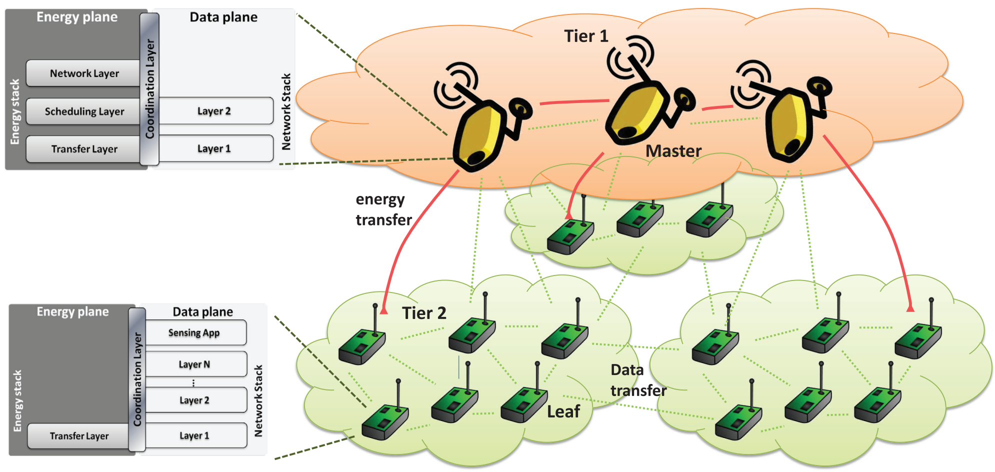


Fig. 2. Network architecture for WET enabled deployments: *masters* on tier-1 are capable of harvesting (or generating) energy and transmitting it towards the *leaves* on tier-2. As opposed to the network stack, the energy stack is more complex in the core of the network and simpler at its edges.

and mitigate line-of-sight constraints by routing energy around obstacles. Similarly, special router nodes, such as the ones installed with reflective mirrors [40] when employing WET technology in the visible spectrum range, can also be deployed at tier-1 to route energy. The *tier-2* consists of in-situ devices (*leaves*) that only harvest and consume energy. Leaves additionally have an energy buffer (e.g., a capacitor) to store harvested energy for potentially uninterrupted operation. While masters assume a superior role in the energy plane, they can act similar to a leaf in the data plane, i.e., sense data and cooperate in its distribution in the network. In other words, they can act as a node in the data routing topology. We build one instance of such a master and a leaf in Section 4.4.1

In this new network model, a notable difference between the energy stack and the traditional network stack is that the latter grows at the edges, whereas the former grows in the center of the network. For example, the core of the data network operates at layer two or three, whereas the leaves, which maintain the application state, extend up to the application layer. In contrast, in the energy plane, the leaves are represented by in-situ sensors that only operate on the transfer layer to receive the energy. The network core, which includes masters in tier-1, also employs higher layers of the energy stack for efficient energy distribution. While similar tiered models have been repeatedly advocated for future WSNs (i.e., IoTs) [4, 14, 15, 29], our proposed architecture makes these efforts more relevant.

3.3 Inter-Layer and Cross-Plane API

We now define the inter-layer and cross-plane API, one of the most important characteristics of a layered software design to express its internal operation in an abstract way. Figure 3 shows a minimal set of runtime APIs[2] for all possible interactions required to drive the functionality of each layer. Please note that below we only define the purpose of the API; its need and the corresponding functionality will be discussed in the subsequent sections. Moreover, the API considers the underlying WET mechanism to be unidirectional. We anticipate minor changes in

[2]Not including the administrative API related to layer configuration.

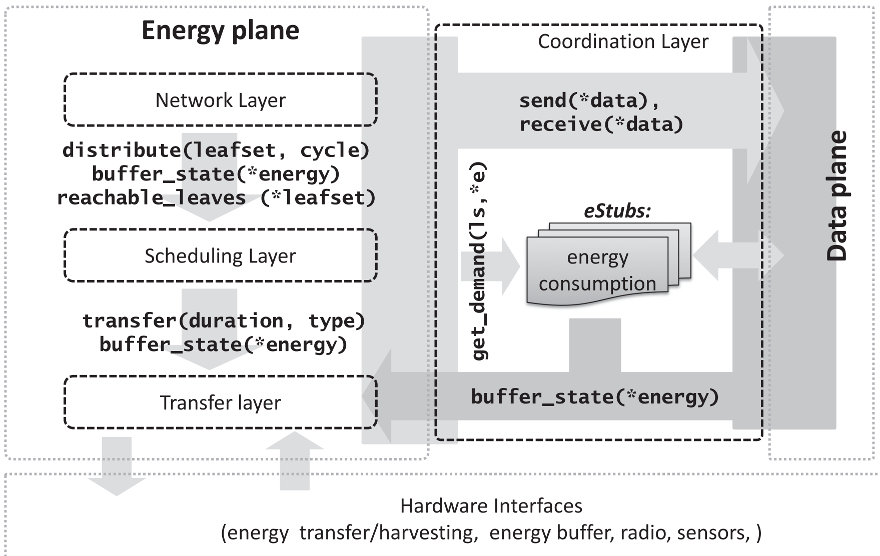


Fig. 3. API between energy layers, and across the energy and data plane through the coordination layer. The direction of arrows depicts the direction of API calls.

the definitions when employing an omnidirectional WET, similarly to how the application API is different for UDP and TCP protocols (further discussion is deferred to Section 8).

- `buffer_state(*energy)`: Returns the energy level of the underlying buffer in `*energy`. This is both an inter-layer and cross-plane API. An energy aware application in the data plane, such as intermittent computing [28], could use this API to make checkpointing decisions.
- `transfer(duration, type)`: Transfers energy for the specified `duration` using a specific WET `type`, allowing support for multiple WET interfaces on a single master.
- `reachable_leaves(*leafset)`: Returns the set of leaves reachable over a single energy hop by a master in `*leafset`.
- `distribute(leafset, cycle)`: Distributes energy to the `leafset` over duration `cycle`. The `leafset` is a subset of *reachable leaves* by a master.
- `send(*data), receive(*data)`: To send and receive control information of any layer in the energy stack using data communication capabilities of the data plane.
- `get_demand(ls,*e)`: Allows scheduling layer to retrieve the energy demand in `*e` for all the leaves in its assigned leaf set (`ls`).
- `eStubs`: These are used to actively estimate the energy requirements of a node (see Section 5.3), for example, by monitoring the radio energy states or by counting packets sent/received and transmiting this information to the stubs at the master for energy scheduling. These stubs are for fine granular energy estimation and management and, hence, optional. An alternative, simpler way for coarse-grained estimates is to measure the energy buffer's depletion rate and calculate a moving average over that.

Use case: To elaborate on the practicality of these API for cross-plane interactions, we use *intermittent computing* as an example of an energy conscious application and illustrate how it could interact with the energy stack for improved performance. Intermittent computing allows successful execution of programs under intermittent availability of power—as would be the case when a single master is responsible to periodically transfer energy to multiple leaves in our network architecture. One method employed by today's solutions [28] is to checkpoint computational state before a power blackout and restore the state at the start of the next activation cycle. However, checkpointing requires expensive writes in the secondary storage, thus minimizing the number of checkpoints is a key performance metric of an intermittent computing solution [5]. Our APIs can reveal necessary energy state at runtime and enable useful interactions to improve the performance of intermittent computing solutions. For example, `buffer_state(*energy)` could offer instantaneous energy state to a checkpointing strategy, allowing it to minimize the number of checkpointing operations by delaying them until the last point in time. Similarly, long-running algorithms in an intermittent computing paradigm can reveal their energy demands to the energy plane through `get_demand(ls,*e)` to avoid frequent checkpointing interruptions. This could, in turn, trigger interactions within the the energy stack layers to update per leaf energy transfer schedules as well as network wide energy distribution.

With the set of API defined, we next expand on their functionality by describing each layer in detail.

4 *TRANSFER* LAYER

The transfer layer of the energy stack is conceptually similar to the PHY layer of a data networking stack. Its main responsibility is the physical transfer of energy from one node to another, i.e., from a master to a leaf. To this end, we identify the following two as the key tasks of the transfer layer:

- To specify the hardware interface of the employed transfer mechanism, and
- The actual transfer protocol used between the master and a leaf for energy transfer.

To develop the transfer layer and realize the overall vision of the energy stack, we first identify an off-the-shelf technology to be used for the physical transfer of energy. Our goal here is not to identify the best among all available WET technologies but rather to select an appropriate technology that can help us demonstrate the feasibility of the proposed energy management solution. We then present the energy transfer protocol administering the physical transfer of energy through the synchronization of masters and leaves.

4.1 Identifying an Energy Transfer Mechanism

Although several recent research efforts propose interesting mechanisms for wireless energy transfer [13, 22, 23, 32, 37], we compare the three most widely used WET mechanisms—radio frequency (RF), laser, light—to handpick a representative technology to build an instance of the transfer layer. We note that our comparison is specific to the off-the-shelf technologies used in our experiments and should not be generalized.

Figure 4 shows that laser μ-power beaming provides consistently good energy transfer at the 80m to 100m range. RF and light transmitters are unable to transfer energy beyond a couple of meters, restricting the range of applications where these mechanisms can be employed. We observe a decrease in laser power transfer at less than 10m due to the laser beam not covering the entire solar panel at these short distances. Another conclusion from Figure 4 is with regard to the class of devices or applications that each transfer mechanism can support. Comparing the three average power requirements for a TelosB mote-class device under different duty cycles, we observe that energy transmission based on light and radio can support 1% duty-cycle operation at a very small range. Laser μ-power beaming can support a duty cycle of up to 10% at a range—comparable to the data transfer range of typical WSN devices.

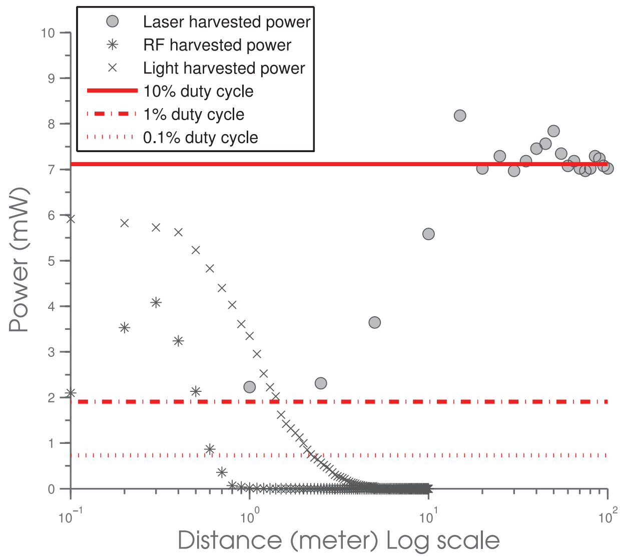


Fig. 4. Harvested power at different ranges (log scale) along with power requirements for TelosB operation at different duty cycles. We use a Powercast module for RF [13], 3W Cree Q3 LED torch [12] for light, and an 808nm 0.8W near-infrared module [36] for laser. For harvesting, we use the P2110 IC [27] by Powercast and a mono-crystalline, high-efficiency solar cell [18] for light and laser.

Overall, we can conclude that laser μ-power beaming, among the three compared, is the most suitable energy transfer mechanism for our requirements to build an instance of the transfer layer: wireless with small footprint, provides energy at radio comparable range, and delivers enough energy to support existing mote class devices. However, we believe that there are other technologies that can be potentially more efficient than lasers. As an example, microwave-based power beaming is less prone to atmospheric attenuation [24].

4.2 Energy Transfer Protocol

After identifying a suitable WET mechanism for our implementation of the energy stack, we now present the energy transfer protocol. On a very basic level, energy transfer simply means directing the transmitter towards a corresponding receiver, e.g., a laser beam shining on a photo-voltaic cell. However, we may still need to synchronize the transmitting and receiving nodes to optimize the transfer operations. For example, a leaf might not be able to accumulate sufficient energy to successfully transmit a data packet if it is continuously leaking energy during the energy transfer operation. We can avoid these cases by synchronizing the nodes to allow a leaf to potentially turn off its radio hardware while harvesting energy.

The transfer protocol, as discussed in Section 3.3, provides a generic API `transfer(uint32_t duration, uint8_t type)`, enabling support for multiple technologies at the transfer layer. The overall operation of the transfer protocol protocol is described in Figure 5. First, the master simply focuses its transmitter (i.e., the laser beam) on a leaf to recharge its energy buffer (e.g., a capacitor). This charging continues until the *Response Interval* (I_{RS}) expires: The leaf should now have have accumulated enough energy to transmit a data packet, marking the end of I_{RS} as well as synchronizing the master and the leaf. The length of I_{RS} is dependent on the state of the energy buffer prior to recharging. After the interval I_{RS}, the protocol repeats in a cycle of variable

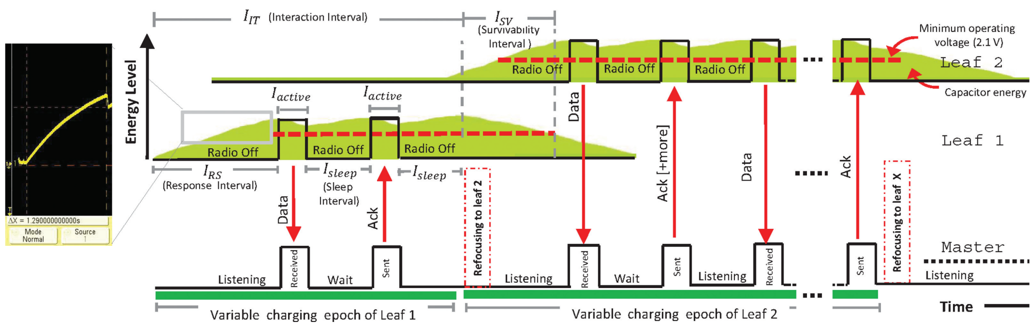


Fig. 5. Energy Transfer Protocol: The master focuses its laser on fully depleted leaves for a charging epoch.

length defined by the following two types of leaf intervals: the optional (i) *Active interval* (I_{active}), to communicate (transmit or receive) data with the master or other leaves thereby facilitating data plane operation, and the (ii) *sleep interval* (I_{sleep}) to accumulate energy while the radio is turned off. We define this variable charging epoch between a master and a leaf as the *Interaction Interval* (I_{IT}). Moreover, the repetition of I_{active} and I_{sleep} intervals determines the *leaf duty cycle*.

The master, after completing interaction with one leaf, will refocus its energy transmitter on another leaf based on the scheduling policy (cf. Section 5). However, after the charging epoch, the first leaf's capacitor retains energy for some I_{SV} (survivability interval), even after the master relocates its energy transmitter. This interval facilitates the operation of a sensing application in the data plane; a leaf can operate continuously with frequent charging epochs that replenish its energy buffer, allowing it to meet its sensing (and communication) requirements.

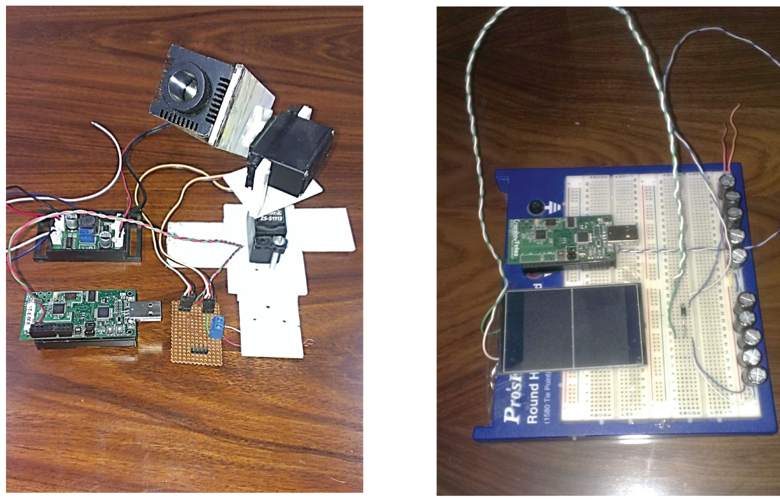
4.3 Supported Communication Modes

We now highlight the two operational modes of a leaf that can be supported in the data plane by the energy transfer protocol. Both these modes offer different advantages while imposing restrictions on performance characteristics such as recharging rate and the number of leaves per master (i.e., scalability), as demonstrated in Section 4.4.4.

Cluster Mode: Leaves only interact with each other through their respective master. Here we use the packets in an I_{IT} to exchange sensed data. This mode is similar to the infrastructure mode in IEEE 802.11 where the hosts can only interact through their access point. The principal advantage of this communication mode is that it improves scalability of the transfer protocol (cf. Section 4.4.4). This improvement is because a leaf can live longer by turning its radio off once the interaction with master is over, allowing the master to serve more leaves before returning.

Mesh Mode: Leaves are free to directly communicate with each other by using the overlap among I_{SV} of leaves. This mode is similar to the ad hoc mode in IEEE 802.11. The principle advantage of this mode is that it enables typical WSN applications requiring a flat, un-tiered deployment of leaves. It is clear that this mode is more power hungry and less scalable.

However, in both modes scalability is only limited if we want an *active network* where all the leaves are recharged before their power levels drop below the operational threshold of TelosB. Otherwise, the scalability of this mode is only affected by application constraints such as the desired recharging and data rates. We provide a detailed quantitative analysis for these different communication patterns in Section 4.4.4.



(a) Master: TelosB with μ-power beaming and pan-tilt

(b) Leaf: TelosB with monocrystalline solar panel

Fig. 6. Hardware implementation of the nodes.

4.4 Evaluating Laser-Based Energy Transfer Layer

Our evaluation of the transfer layer answers three fundamental questions: (1) *What is the maximum duty cycle supported by the energy transfer protocol*?, (2) *What kind of precision is required when pointing a laser source to a leaf*?, and (3) *What is the scalability of the transfer layer in terms of the number of leaves per master when using a laser-based WET*? Before answering these questions we briefly describe the hardware design of master and leaves used in our evaluation.

4.4.1 Hardware Design. Figure 6 shows the hardware implementation of master and leaf. A master (cf. Figure 6(a)) is equipped with a laser module. This module provides the energy transmission capability and is mounted on a dual-axis pan-tilt mechanism based on two Futaba s3003 servo motors controlled from a TelosB node using PWM enabling a 180° rotation on both axes. The master uses the TelosB radio for data communication with leaves.

A leaf (cf. Figure 6(b)) is also a TelosB node equipped with a monocrystalline photo-voltaic cell, which can receive energy from a focused laser beam. Between the photo-voltaic cell and the TelosB input, we additionally attach a capacitor to allow energy buffering, thus temporally decoupling energy reception and computation. We implement the transfer layer protocol in TinyOS.

4.4.2 Duty Cycle and Data Rate. A first important question we want to answer is about the data rate during master–leaf interaction and the largest leaf duty cycle supported by the transfer layer. The duty cycle in this particular scenario is the duration for which the radio hardware of a leaf remains active during I_{IT}.

An answer to both these question helps understand the range of WSN applications that can be supported by our system. Using laser μ-power beaming mechanism, we expect to support a duty cycle of around 10% (cf. Figure 4). We vary I_{sleep} to alter the leaf's duty cycle. Figure 7 shows that a laser-based transfer layer can provide a maximum data rate of 350Bps at a duty cycle of 7.4%, coming quite close to the 10% we expected after our power harvesting experiments in Section 4.1. We empirically observed that this small difference is due to harvesting inefficiencies of TelosB, since its resistance is not optimized for the solar panel.

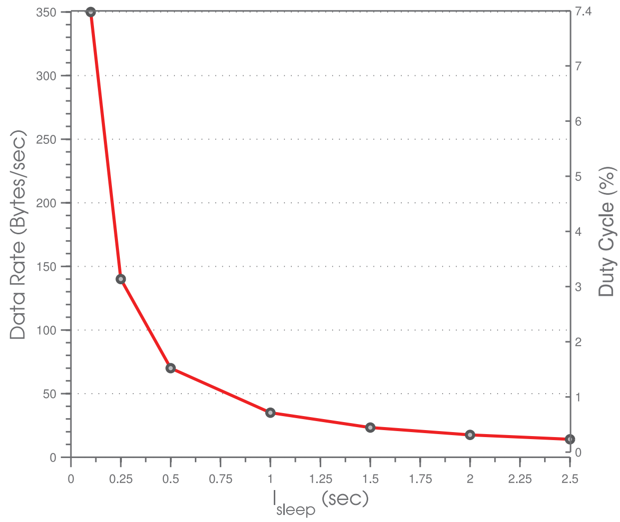


Fig. 7. Maximum leaf duty cycle supported by the transfer layer.

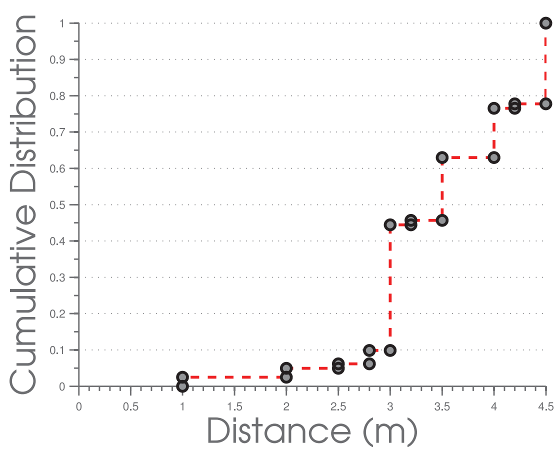


Fig. 8. Re-calibration of master: CDF of servo motor error at different distances.

4.4.3 Calibration Requirements. For recharging the leaf, we need to ensure that the laser-beam correctly points at the solar panel of the leaf. The master might require to re-calibrate once it returns to the same leaf after recharging all other assigned leaves. The frequency and the magnitude of this re-calibration strongly depends on the movement-error characteristics of a particular servo motor.

We measure this error for our servo motors by fixing the laser beam at a certain point, i.e., the origin, before moving it a complete 180° along one axis and then back. After each iteration we manually measure (with a metric rule) the radial distance from the origin. We repeat this experiment 100 times. Figure 8 shows the CDF of these errors from the origin. We can clearly observe that the errors are *bounded* with a maximum radial error of 4.5cm, preventing cumulative error build up. Also, nearly 90% of all error values lie between 3 and 4.5cm. During our experiments, we also observe that these errors do not self-cancel as they remain in the same quadrant.

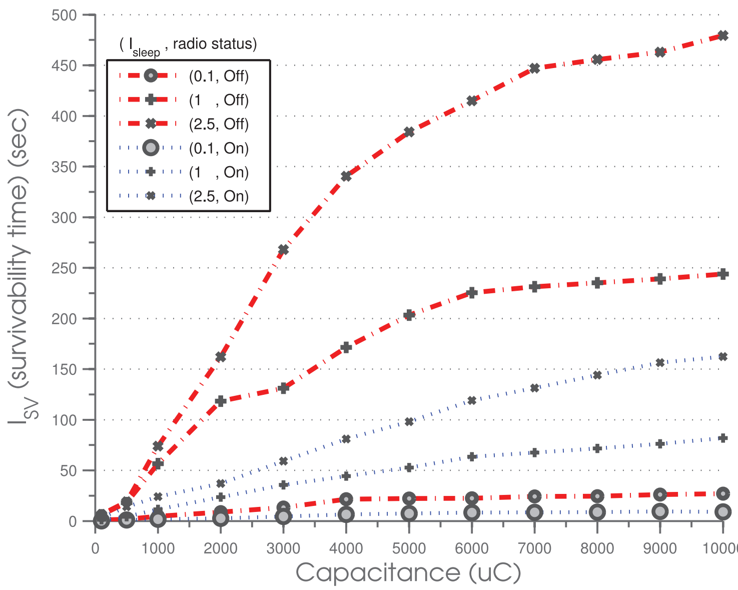


Fig. 9. Survivability: I_{SV} as a function of I_{sleep} (for recharging) and capacitance (charge capacity).

Thus, for a leaf node with 8.8×5.4 cm solar panel and a perfectly centered laser beam, in the worst case, the master will have to recalibrate after every round trip if the master-leaf distance is greater than 29.5m.

Technological Limitations: We emphasize that these error bounds are for a particular motor technology, i.e., Futaba's s3003. Hence, these results should not be generalized to define the accuracy limits of laser-beaming used in an energy distribution architecture. The selection of a motor technology is a tradeoff between its cost and the required re-calibration effort for a particular architecture. Since re-calibrating the master is not within the scope of this prototypical study, our choice is biased towards the overall cost of nodes.

4.4.4 Scalability. We define the scalability as the number of leaves a master can energize perpetually. Since energy distribution among multiple leaves is the responsibility of the scheduling layer, which we discuss in the upcoming Section 5; here we assume a simple scheduling layer that energizes leaves round robin. The scalability is strongly dependent on I_{SV}, which is a function of the charge retained by the capacitor even after the master has refocused to another leaf. Thus, we first empirically observe the impact of different factors on I_{SV}. We then describe the mathematical model[3] employed to determine the scalability of our system.

Survivability Limits for Leaves. I_{SV} is a function of three parameters: The capacitance of a capacitor, I_{sleep}, and how often radio hardware, which dominates the energy consumption of a TelosB node, is used. We measure this time empirically by observing when the capacitor voltage drops below 2.1V (the minimum operational voltage for TelosB), while we vary these parameters. Figure 9 depicts the result. We observe that increasing I_{sleep} significantly improves I_{SV} when using

[3]Our experimental deployment is limited to four leaves.

larger capacitors in both on and off radio states. Thus, we see greater survivability only when a capacitor is large enough to store the excess energy collected over a longer sleep interval.

Modeling the Recharging Rate. We observe that a given survivability time can be utilized to either increase the number of leaves (N) that can be served or alternatively reduce the recharging rate (RR), that is, how often a master has to return to charging the same leaf. To understand the tradeoff between these two variables, we model a network where leaves are kept equidistant. The following equation then defines RR based on the interaction time (I_{IT}), the number of leaves (N), and $T(X)$ which is movement time required by the servo motor to move X degrees between leaves:

$$RR = N \times I_{IT} + 2(N-1) \times T(X). \tag{1}$$

Based on the description of I_{IT} in Section 4.2 and a graphical representation in Figure 5, it can be computed as follows:

$$I_{IT} = I_{RS} + (I_{active} + I_{sleep}) \times K, \tag{2}$$

where K is the number of repeated cycles of I_{active} and I_{sleep} during a single charging epoch. If we want to maintain a continuously *active network*, then the master must return to recharge each leaf before its respective power level falls below 2.1V, which, by definition, happens after I_{SV}. Note that for such a scenario, $I_{RS} = 0$, thereby significantly reducing the interaction interval with each leaf. To maximize the number of leaves supported in any setting, we make two assumptions: First, since a larger RR supports a greater number of leaves, we use the largest possible value of $RR = I_{SV}$. Second, we also keep all leaves separated by 0.49°; the minimum separation allowed by the servo motors used in the master. Plugging the value of $RR = I_{SV}$ from our experiments (Figure 9) and the constant value for $T(X) = 1.85$ms as the movement time for the servo allows us to determine the recharging rate for a particular number of leaves per master for different configurations of I_{IT}.

Results. In an active network, the communication mode (cf. Section 4.3) greatly impacts the duration of I_{SV}. We therefore separately evaluate both the mesh and clustered mode of operation.

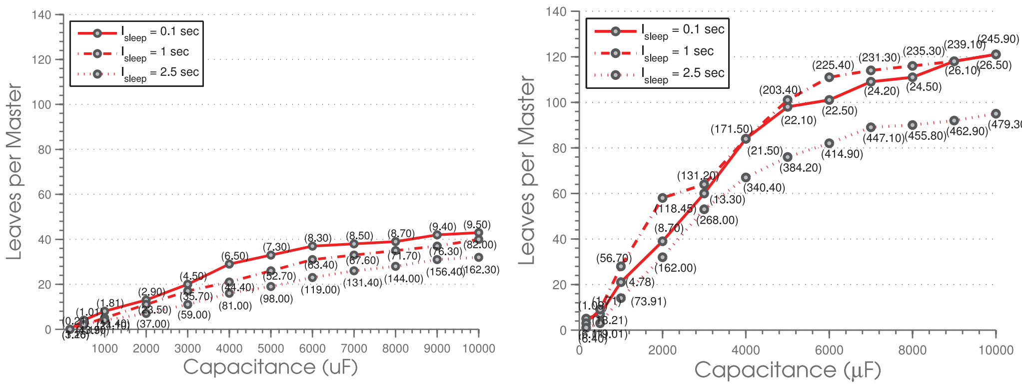
Recall that for **mesh operation**, all leaves use their radio for direct communication with each other even after the master relocates its laser beam. While any radio duty cycle can be supported, for simplicity we continue with the duty cycle determined by I_{sleep} for our evaluation. We notice that the network can scale up to 40 leaves per master with a recharging rate of 9.50s in this mode (Figure 10(a)). This implies that for a recharging rate of 9.5s, with a 10mF capacitor, we can *perpetually* energize a 40 node, full-mesh, WSN.

We next look at the **cluster operation**, where leaves only *sense* at an application defined rate with no intra-leaf communication after the master relocates its laser beam (all data routing occurring through the master during a charging epoch). With lesser energy consumption we expect it to achieve greater scalability than mesh operation. Figure 10(b) validates this as the network can now scale up to 120 leaves per master with a recharging rate (or polling rate) of 26.5s. This implies that for a recharging rate of 26.5s, again with a 10mF capacitor, we can *perpetually* energize a 120 node WSN that networks only via masters.

Overall, we can conclude that a laser-based transfer layer scales well for practical WSN deployments.

5 SCHEDULING LAYER

The transfer layer specifies the details of physical energy transfer between a master and a leaf. With this transfer capability in place, the next question is how do we schedule such energy transfers: Given a set of leaves, the master node has to decide *how often, how much, and to which leaf is the energy to be transferred?* The answers to these questions are trivial if the amount of energy that can be transferred over a certain period by a master "always" exceeds the requirement of the *leaf*



(a) Mesh: Radio use between leaves (b) Cluster: Radio use only during master-leaf interaction

Fig. 10. Scalability: While maintaining an always-on network, laser-based transfer layer scales up to 40 and 120 leaves per master in mesh and cluster modes, respectively. The marker values represent the recharging rate in seconds.

set—the set of leaves energized by a single master. Assuming that the master knows the energy demand of each leaf in its leaf set, it can simply serve individual demands round robin.

However, the answers become nontrivial as demand exceeds supply or when energy demands are unknown. This nontrivial case may occur, for example, when the master itself is bounded by a finite supply of energy or if the time required to recharge the whole set of leaves exceeds the desired recharging rate of an individual leaf. In such situations, the scheduling policy is a decisive factor in optimizing network operation along a given performance axis. WET, as opposed to natural energy harvesting, provides network designers with the unique opportunity to manage energy supply based on application requirements. As an example, some applications might prefer data fidelity (sensing depth) over deployment coverage (breadth). This could be achieved by prioritizing energy supply to fewer leaves with higher sensing and communication rates. This section tackles the nontrivial case and explores different scheduling algorithms to determine their efficacy for a given network level optimization goal.

5.1 Scheduling Responsibilities

The scheduling layer of the energy stack is responsible for distributing energy among the leaves in the leaf set of a master node. This is conceptually similar to layer 2 of the OSI model, which is responsible for sharing the medium and thereby the transmission bandwidth between nodes in single hop distance using multiplexing techniques such as time, space, code, or frequency division. While efficiently scheduling energy transfers could also be required in omnidirectional WET, for example, to synchronize it with energy availability at the master, it is a fundamental issue in directional WET where a master can only recharge one leaf at a time. We identify the following as key responsibilities of the scheduling layer:

- Localization of leaves when using a directional energy transfer mechanism,
- Periodical collection of the leaves' energy requirements assuming the master can be reached by each leaf either through a direct or indirect data link, and
- The identification and use of the best scheduling algorithm based on high level, potentially application-specific network optimizations.

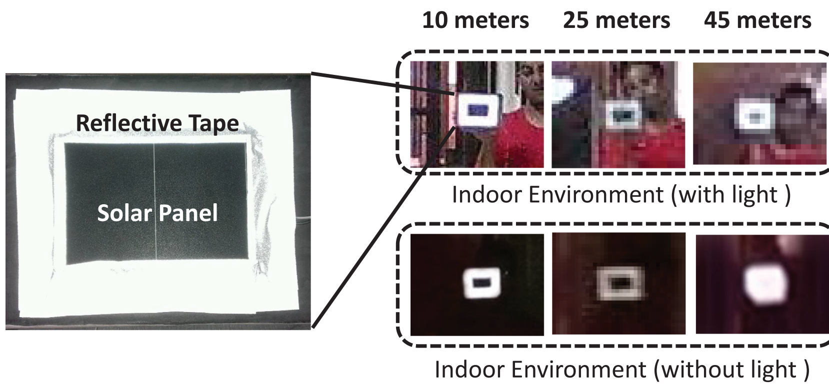


Fig. 11. Leaf localization: Reflective tape reflection intensity at three different distances.

Especially when employing a directional energy transfer mechanism, e.g., a laser, a master node needs to be aware of a leaf's location. In the following, we therefore first describe leaf localization. We then formally describe the scheduling problem and show how the energy requirements of each leaf in the leaf set are collected. Finally, we present a set of candidate scheduling algorithms.

As opposed to the transfer layer, where we had to choose one WET mechanism to implement the layer, here we just explore the feasibility of candidate algorithms rather than selecting one. This is mainly because the choice of the scheduling algorithm could be driven by the requirements of a given application. We compare these algorithms and establish their effectiveness using general network performance measures such as data fidelity and deployment coverage. Our worst-case benchmark for this comparison is round-robin scheduling.

5.2 Leaf Localization

Localization is required when the transfer layer employs directional WET mechanism. It is a well-known research problem in WSN [2, 17, 20], but almost all of the proposed solutions require the nodes to actively participate. This assumption is not valid for our system where only the master possesses energy, while the leaves are energy deficient. A master needs to know the physical coordinates of all leaves to be able to provide energy to the individual leaves. Depending on the exact deployment characteristics, we consider two possibilities for this purpose: dynamic localization with a search phase in the beginning or static preassigned locations known at boot time.

Dynamic localization is possible using a fine-grained search of the region assigned to each master. During this search, the master can continuously transmit energy along grids in its assigned region. As a leaf with a power measurement circuit can determine the received power, it reports itself to the master whenever the received power crosses the ambient power threshold. The master can use this positive feedback to determine and fine-tune the coordinates of a particular leaf. Alternatively, an image processing–based approach can also be utilized. In this approach, a master is additionally equipped with a camera and flash, while the leaves have their solar-panel with a reflective tape on its edges as shown in Figure 11. This reflective tape is visible even at a distance of ≈45m to 60m when light is directed on it. Thus, when the camera captures an image with flash, multiple nodes can be identified due to reflection from the tape and their coordinates can be calculated.

The second, simpler approach considers a static deployment, where the coordinates are manually provided to the master. The master then uses these coordinates to deliver energy to leaves based on the scheduling strategy.

While we already implemented support for searching in our prototype, allowing dynamic localization, we believe that optimizing the searching algorithm dealing with the practical concerns of the searching is a research problem in its own right and part of future work. Hence, in this article, we use the second approach to focus on evaluating the scheduling itself.

5.3 The Scheduling Problem

Each master m in the network is intended to provide energy to its leaf set $ls_m = \{l_1, l_2 \dots l_n\}$, i.e., a subset of leaves in the network that can be recharged directly by this master. The amount of energy transferred to leaf l_i is dependent on its workload w_i which is the sum of three tasks: the number of packets sent (p_s) and received (p_r) and the number of sensing activities (s) performed [4] during a time period t_{cycle}. The value of t_{cycle} can be appropriately chosen based on the sensing and communication rates in the network. Thus, the total energy required to perform w_i is e_{w_i}:

$$e_{w_i} = e_{p_r} + e_{p_s} + e_s. \tag{3}$$

If the time required to supply this energy is t_i, then the total time to serve the energy requirements of ls_m is

$$T_{ls_m} = \sum_{i=1}^{n} t_i. \tag{4}$$

The trivial case, i.e., a master that can provide sufficient energy to all leaves in ls to perform their workloads over the period t_{cycle}, is

$$T_{ls_m} <= t_{cycle}. \tag{5}$$

However, here we focus on the nontrivial case, i.e., if the master cannot fulfill energy requests of all nodes ($T_{ls_m} > t_{cycle}$). For efficient scheduling of energy, each leaf periodically communicates its w_i to m using the coordination layer (cf. Section 3.3). Note that our key metric to quantify a leaf's share of energy is in terms of t_i, as we do not restrict the energy supply at the master but rather the time it takes to fulfill the demand of each leaf. The calculation of t_i is carried out individually for each leaf depending on its distance from the master and the characteristics of the medium between them. Alternatively, a master may calibrate itself by monitoring a leaf's recharging characteristics.

The scheduling layer simply receives its leaf set and the length of t_{cycle} in milliseconds from the network layer (API: `distribute(leafset ls*, uint16_t cycle)`) and then uses a scheduling algorithm to distribute energy. We defer a more detailed discussion on why the leaf set is provided by the network layer to Section 6.

5.4 Scheduling Algorithms

There are two facets of a resource sharing model for energy supply in WSNs. The first one deals with efficient distribution of energy by a master to its leaf set. The second one deals with adapting a leaf's operation locally based on the proportion of energy demand met by its master node. In this article, our main focus is the efficient distribution of energy. A number of solutions [8, 31] have been recently proposed for adapting a node's duty cycle based on its harvested energy. The integration of these solutions with the proposed system architecture to further optimize network operation remains a future work.

Once a master knows the energy requirements of its leaf set, the next step is to adopt a resource sharing regime for energy distribution. The simplest, most homogeneous approach is round robin scheduling which uniformly distributes the same amount of energy to all leaves regardless of their respective workloads. However, in a network with non-uniform workloads, there is a great chance

[4]For brevity here we ignore the additional computational overhead.

ALGORITHM 1: Max-Min Fairness

Data: $\{t_i : l_i \in ls_m\}, t_{cycle}$
Result: $\{CT_i\}$ // CT_i: charging time assigned to each leaf l_i
$T_a \leftarrow \{t_i : \forall l_i \in ls_m\}$ // T_a: the working set of charging times to assign
$CT_i \leftarrow 0$
while $t_{cycle} > 0$ **do**
    $t_{min} \leftarrow \min T_a$
    for $l_i \in T_a$ **do**
        $t_i \leftarrow t_i - t_{min}$
        $t_{cycle} \leftarrow t_{cycle} - t_{min}$
        $CT_i \leftarrow CT_i + t_{min}$
    end
    $T_a \leftarrow T_a \setminus \{l_i : t_i = 0\}$ // Remove all zero entries from T_a
end

of energy being wasted if a node is unable to take advantage of its allocated energy. In addition, in an application-specific WSN the *scheduling goal* might not be uniformity but meeting specific application requirements. We therefore propose the following three resource sharing algorithms.

5.4.1 Max-Min Fairness. Max-min fairness is a popular statistical resource sharing model also used for communication link capacity assignment. The main idea behind max-min fairness is to serve small demands, dropping the ones from contention that are met, and let the larger demands split the rest of the resource. In case of equal demand, round-robin scheduling is max-min fair. Algorithm 1 shows the max-min logic at the master node *m* for energy scheduling.

We believe that max-min fairness has the potential to perform well and meet certain general WSN application requirements. For example, since it equally distributes energy in the network until all small demands are met, we expect it to achieve better coverage. However, since it emphasizes fairness, it might not be an ideal choice for deployments where each node is not equally important. For example, in a collection tree, which is the most common communication scenario by far in multi-hop WSNs, the nodes higher up in the topology must be provided with more energy. These nodes are not only required to report their own sensing values but also have to forward the data from their descendants in the collection tree. Hence, if these nodes are not prioritized for energy distribution, max-min fairness could adversely effect coverage as well, e.g., as forwarding fails due to too little energy.

5.4.2 Weighted Max-Min (WMM). To address the limitations of max-min fairness, we modify it by assigning weights to each demand. This enables the energy scheduler to prioritize leaves with larger energy demands.

Each participating demand is assigned a certain weight. The allocation starts by assigning a unit resource to the node with the highest weight. In the subsequent rounds, the weight of previously served nodes is decremented by the unit resource possibly including more nodes in the contention. The process continues until the weights become zero. The assignment of weights is open for manipulation and hence in its most rigorous version WMM has the potential to reverse the max-min logic altogether by assigning significantly higher weight to larger demands. In such cases WMM will serve larger demands first and *once they are met*, the smaller demands split the rest of resource. We believe that WMM (cf. Algorithm 2) provides much needed liberty to WSN developers to enforce application requirements.

ALGORITHM 2: Weighted Max-Min

Data: $\{t_i : l_i \in ls_m\}, \{w_i : l_i \in ls_m\}, t_{cycle}$ // t_0: the unit amount of time
Result: $\{CT_i\}$ // CT_i: charging time assigned to each leaf l_i
$T_a \leftarrow \{t_i : \forall l_i \in ls_m\}$ // T_a: the working set of charging times to assign
$CT_i \leftarrow 0$
while $t_{cycle} > 0$ **do**
  $i \leftarrow \arg\max_i \{w_i : l_i \in T_a\}$
  $t_i \leftarrow t_i - t_0$
  $w_i \leftarrow w_i - 1$
  $t_{cycle} \leftarrow t_{cycle} - t_0$
  $CT_i \leftarrow CT_i + t_0$
  if $t_i = 0$ **then**
    $T_a \leftarrow T_a \setminus \{l_i\}$
  end
end

ALGORITHM 3: Demand-Supply Fairness (DSF)

Data: $\{t_i : l_i \in ls_m\}, t_{cycle}$
Result: $\{CT_i\}$ // CT_i: charging time assigned to each leaf l_i
$Tl_m \leftarrow \sum_i t_i$
for $l_i \in ls_m$ **do**
  $CT_i \leftarrow \frac{t_{cycle}}{Tl_M} \cdot t_i$
end

If weights are carefully chosen, then this algorithm has the potential to balance both deployment coverage and collection performance. In simpler network topologies, such as a single hop star, WMM can provide maximum data fidelity (sensing depth) by prioritizing nodes with the highest sensing and communication rates as we show in Section 5.6.1.

5.4.3 Demand-Supply Fairness (DSF). Our final algorithm (cf. Algorithm 3) is fair to the ratio *demand* : *supply*. Here the algorithm takes as an input the total resource and all the demands, and then computes a fixed percentage of resource that can be assigned to each demand: If the computed fraction is p, then each node gets the p percentage of its own demand. In other words, this assignment globally minimizes the *demand* : *supply* ratio. A very careful assignments of weights in weighted min-max could yield identical results to demand-supply fairness, but we use this algorithm as it functions independent of any *weight inputs* and requires separate discussion and evaluation.

DSF balances the energy supply in a network with nodes of varying responsibilities and importance. We expect it to achieve a sensible performance in terms of both sensing depth and breadth. This algorithm could be used, for example, to bootstrap a multi-hop deployment. With further refinements, it may be applied as the application requirements evolve.

5.5 Evaluation Methodology

Unlike the transfer layer, where the physical characteristics of a particular WET technology is the primary concern, the higher layers are more algorithmic in nature. Our five node deployment is not sufficient to fully evaluate these algorithms. We need a more controlled and repeatable

Table 1. WET Models Based on Empirical Dataset

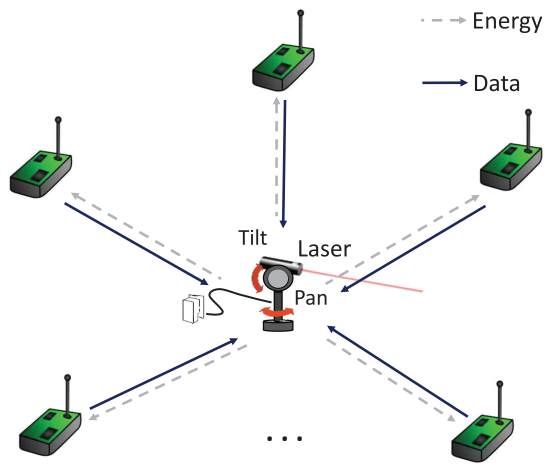
Component	**Model Input** (in evaluation)
Transmitter	laser signal energy in watts (0.8W)
Movement	mechanism: servo motors controlled pan/tilt
	minimum angular movement: in degrees (0.49°)
	movement delay: in ms (3.78ms per 0.49°)
Propagation	laser propagation traces from Reference [6]
Harvester	monocrystalline photo-voltaic cell
	efficiency: ≈20%
Buffer	capacitor in μF (100μF)
	charging: standard RC circuit ($E = 0.5CV^2$)
	discharging: radio load—Idle, Tx and Rx [26]

evaluation environment. However, from our experimental deployment, we gain sufficient knowledge about the physical characteristics of the transfer technology to build a model and integrate it into a simulation framework for further evaluations. Our evaluation methodology for the scheduling layer thus uses a simulation framework [1] built around the TOSSIM simulator [21], which is parameterized with the laser model obtained from the empirical data [6] as summarized in Table 1. The model precisely characterizes the transmitter and its movement control mechanism (cf. Section 4.4.1), ensuring that the characteristics of the pan-tilt mechanism with respect to the minimum angular movement and delay are met. Similarly, the laser propagation and harvesting model is entirely based on empirical traces [6]: The path loss (or attenuation) is dependent on the medium characteristics and the distance between the source and the receiver, both input via a configuration file. Finally, the energy buffer is modeled as a capacitor. The charging rate of a capacitor is faster at the start and then decreases as the capacitor takes on additional charge at a slower rate depending on the time constant τ. The discharging rate depends on the applied load, i.e., energy consumption model. For our initial evaluation, we merge the energy consumption model of the CC2420 radio chip and the sensors [26] of a TelosB node. On top of this framework, we implement the transfer protocol described in Section 4.2.

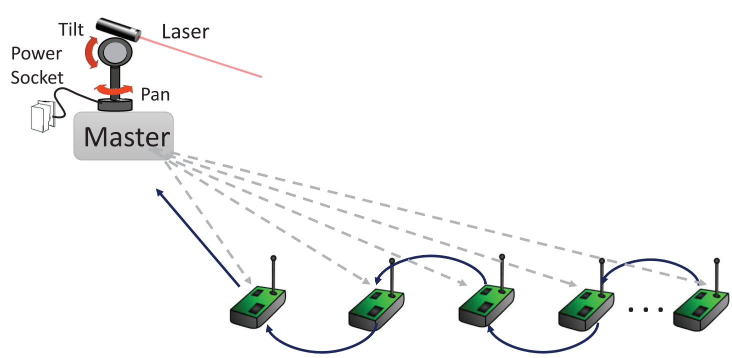
5.6 Evaluating Scheduling Algorithms

To evaluate the scheduling algorithms, we use the two extremes of a tree as representative topologies, star and linear (see Figure 12). In star topology all the leaves are radially spread from the master. In the linear topology, each upstream neighbor forwards its own measurements as well as the packets of downstream neighbors towards the base station. Our evaluation is focused on two metrics, the packet reception rate (PRR) and the deployment coverage. PRR calculates the total number of data packets received at the base station regardless of where these packets came from. The deployment coverage measurement also takes into account the uniqueness of these packets, that is, how many leaves could successfully deliver a minimum number of packets. For our comparison, we use two types of benchmarks, the best and the worst case. Our best case is represented by network performance without any energy constraints. Hence, we run our simulations without specifying energy constraints and record the results. Our worst case benchmark is round robin scheduling, a trivial solution to the scheduling problem, to show if and how the proposed algorithms can improve on it.

5.6.1 Task Completion on Individual Leaves. Before presenting the overall result for PRR and deployment coverage, we first analyze the performance of each individual leaf under different



(a) Star topology

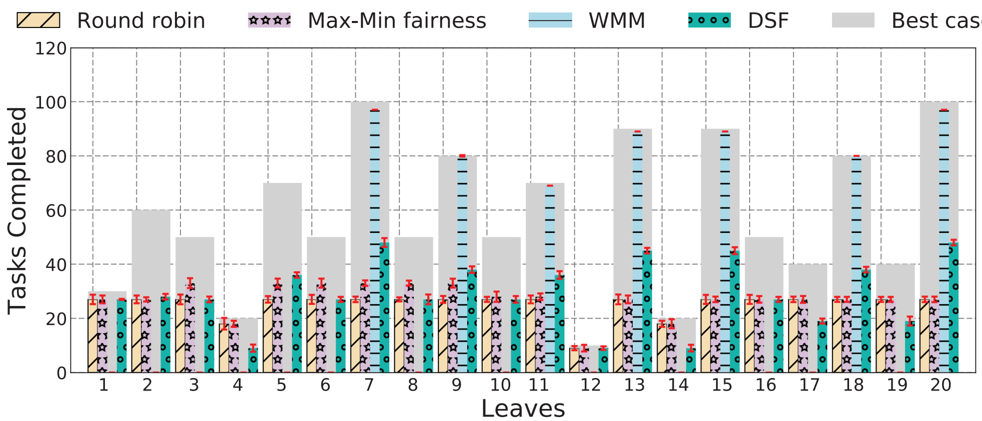


(b) Linear topology

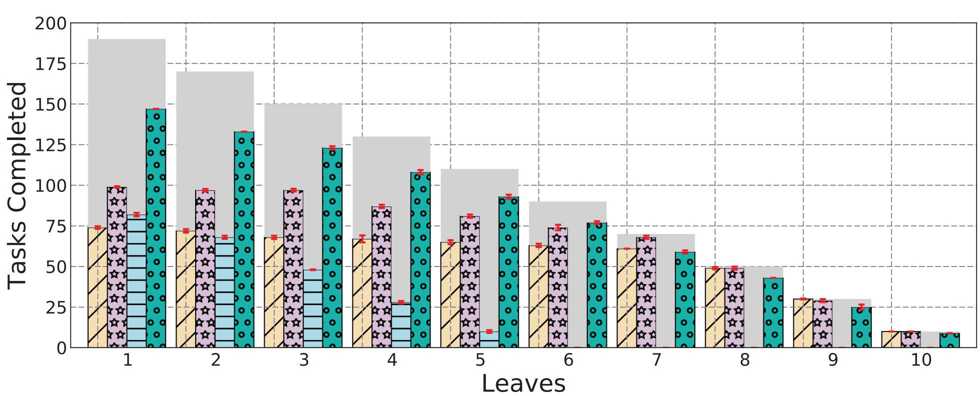
Fig. 12. Exemplary topologies used for evaluating scheduling algorithms.

scheduling algorithms. For this purpose, we calculate the number of tasks completed by each individual leaf in both topologies, with a task being either the transmission or reception of a packet. Both these radio actions consume almost the same amount of energy on TelosB [26].

Star Topology. We simulate a star topology with twenty leaves, sufficient for comparing scheduling algorithms in a single hop scenario. This topology mimics the cluster mode communication, as discussed in Section 4.4.4, where a master acts as a cluster head and leaves only communicate with each other through the master. Each leaf randomly generates its workload w. During workload generation, we ensure that we are in the non-trivial case, i.e., $T_{ls} > t_{cycle}$. Figure 13(a) shows our results for the star topology, where the bar represents the average and error bars represent the standard deviation. The minor differences in repeated simulation results are due to variable packet loss. Although we set the parameters to minimize the packet loss and focus on the energy scheduling artifacts only, the TOSSIM simulation employs a lossy communication model in which some packets will be lost in any case. The y-axis shows the number of tasks completed. We see that the round robin algorithm achieves almost a constant number of tasks (≈25) completed across all leaves. In our test case, this resulted in energy being wasted at leaves 4, 12, and 14 as these



(a) Star topology



(b) Linear topology

Fig. 13. Number of tasks completed by each individual leaf when using different scheduling algorithms.

leaves receive more energy than their demands. Max-min fairness completes the same number of tasks as round robin. This is because in the case of equal demand or when demand exceeds supply on all leaves, which is the case in Figure 13(a) except for the three aforementioned leaves, round-robin scheduling is max-min fair. WMM completes a higher number of tasks at nodes with exceedingly larger workloads (leaves 7, 9, 11, 13, 15, 20). Since we assign very high weights to leaves with larger workloads, we see that WMM is unable to supply any energy to all other leaves with smaller workloads. Finally, the number of tasks completed on a certain leaf when using DSF is dependent on the magnitude of workload on that leaf. We see that each leaf is able to complete a constant percentage of its expected workload, i.e., commensurate to the energy supplied based on minimizing the demand vs. supply ratio.

Linear Topology. To evaluate a linear topology as the other extreme of a tree topology, we simulate a 10-hop network where the base station is located at one end of the topology. An upstream leaf l_i, besides forwarding its own sensor readings, has to forward all the data it receives from its downstream neighbor l_{i+1}. This topology makes use of the mesh mode (cf. Section 4.4.4) where

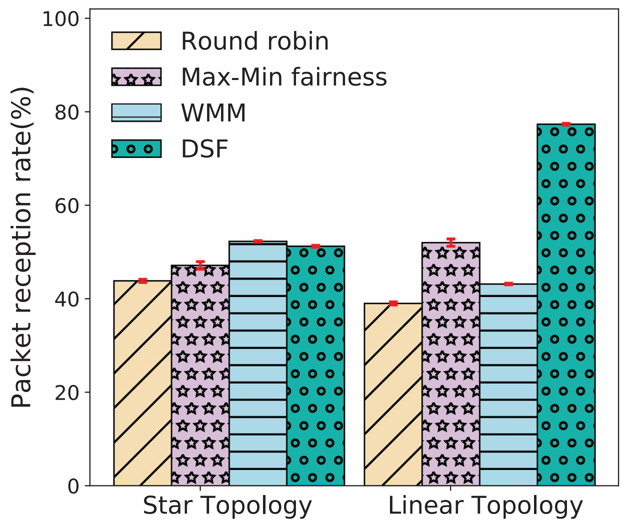
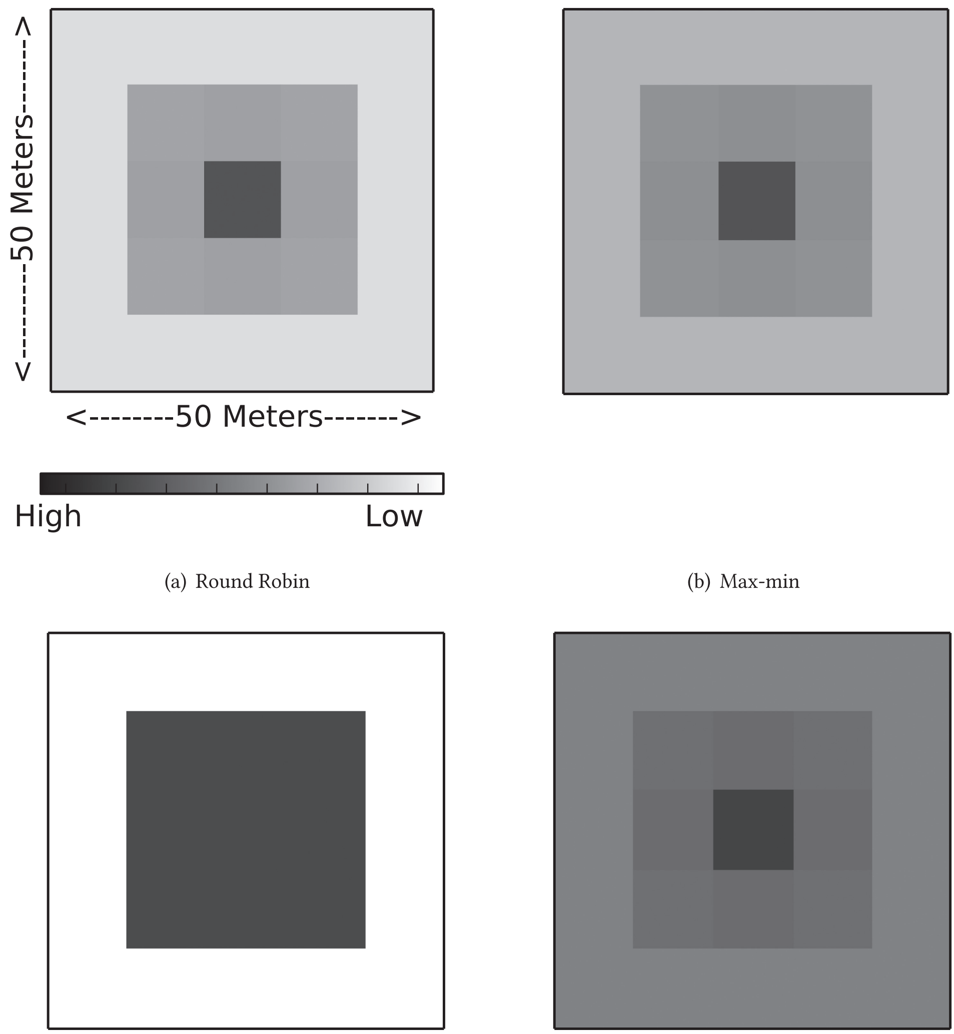


Fig. 14. Packet reception rate at the base station.

leaves can directly communicate with each other. As in the case of the star topology, a single master powers all the leaves. As opposed to the star topology, where we randomly generate workloads, here, each node is assigned a constant workload over a certain time period. However, the *expected* workload (or the number of tasks) of a given leaf depends on the number of descendent leaves. In Figure 13(b), we can see that round robin results in energy being wasted at the rare end of the topology where leaves have small workloads. This results in reduced task completion as nodes closer to the base station are unable to get sufficient energy to deal with their high workloads. Max-min improves on round robin as it avoids energy wastage. However, the energy distribution mechanism still does not favor the nodes closer to the base station and hence performance suffers with the same predicament as with round robin. However, an important observation is the negative impact of WMM on the performance in a multihop topology: Since it provides maximum energy to nodes closer to the base station, the task completion of these nodes suddenly drops. The reason for this is that descendant nodes are not powered, thus more upstream nodes do not receive enough packets from their descendant nodes. DSF strikes an efficient tradeoff between max-min and WMM: By providing some energy proportional to a leaf's workload, it allows leaves at the rare end to generate packets as well as favors leaves closer to the based station to accomplish more tasks.

5.6.2 Network Performance. After analyzing individual leaves, we now discuss the two network level performance measures: PRR and deployment coverage. WMM achieves the highest PRR for a star topology as shown in Figure 14. This is mainly because it favors the leaves with the highest packet sending rates while the workload of individual leaves is independent of each other. However, DSF clearly achieves the highest PRR in the case of linear topology with max-min fairness a distant second (cf. Figure 14). To evaluate the network coverage achieved by these algorithms, we simulate a network of twenty five leaves placed in a grid like topology of 50×50 meters as shown in Figure 15. The grid has 25 cells where each leaf occupies the middle of its cell and the base station is placed in the center of the grid. The communication range is restricted to adjacent cells only. The nodes thus create a tree topology to forward the packets to the base station. We can see that the coverage of round robin and max-min fairness spans across the whole deployment but the



(a) Round Robin (b) Max-min

(c) WMM (d) DSF

Fig. 15. Network coverage in a grid deployment with base station placed in the center.

fidelity of data (represented by the darkness of the cell) is less compared to the DSF. This happens because nodes closer to the base station do not get sufficient energy to forward the packets from nodes located in the outer most cells. Since DSF minimizes the supply demand gap, it outperforms all other algorithms in a tree topology both in the terms of data fidelity (PRR) and deployment coverage.

6 NETWORK LAYER

The scheduling layer decides how a master can schedule energy transfer to leaves in its leaf set. However, any energy transfer optimizations at the scheduling layer may not be sufficient to optimize network wide performance. As an example, consider a linear topology of n leaves powered by two masters where each leaf can only communicate with its adjacent neighbors. Without a global view of such a network, one logical solution may be to assign $n/2$ leaves to each master. This may result in a huge energy deficit at the master responsible for energizing nodes closer to the base station even though the available energy at the other master may be under-utilized because of the low workload of its leaf set. This problem can be resolved, for example, by dynamically changing the leaf set of masters or by enabling energy sharing (possibly over multiple hops) among them. However, such a functionality can only be useful if we know network wide supply–demand gap at each master, that is, a global energy view of the network.

6.1 Networking Responsibilities

The network layer in our proposed network architecture is meant to create a global energy view and thereby balance the network wide energy distribution. Recall that in our tiered network architecture, the network layer comprises more capable nodes in the tier-1 called masters. These nodes are responsible for generating energy, perhaps from a renewable ambient source or using mains power, and for transferring this energy to leaves. Overall, we identify the following key responsibilities for the network layer of the energy stack:

- Periodic sharing of demand vs. supply information among masters to create a global energy view of the network.
- Update the leaf set of each master to achieve better demand-supply balance.
- Sharing of energy among masters, possibly over multiple hops.
- Routing energy across obstacles when using line-of-sight-restricted WET.

Similar to how routers share routing tables with each other, the network layer of the energy stack employs periodic sharing of demand-supply information among masters. Given that the masters in our tiered network architecture are more capable with meagre energy constraints, this additional network traffic for sharing vital energy information is likely not a major concern. This information can help all the masters in tier-1 to create a global energy view of the network which can be utilized in the following ways:

(1) If the demand-supply gap between at a certain master has increased significantly, then the network layer can decide to change the leaf set membership of a subset of its leaves. This, however, is only possible if the corresponding leaves can be reached by their new master. The set of such *overlapping leaves*[5] can easily be determined using localization procedures discussed in Section 5.2. We also note that the reach-ability of leaves through multiple masters inherently supports fault tolerance, i.e., if a certain master is damaged, then its leaf can be reassigned to other masters.
(2) Masters can share energy among each other as they are capable of both transferring and harvesting energy. However, such an approach must also take into account the energy losses incurred (over multiple hops) due to repeated energy conversion cycles. An alternative approach could employ larger harvesting units deployed at locations with abundant harvestable energy, later redistributed among master through WET.
(3) For directional WET mechanisms with line-of-sight requirements, such as lasers, either masters can be equipped with mobility support or special router nodes can be employed with energy mirroring capabilities. Although such special nodes create new network layer challenges, previous work has established its basic feasibility such as in outdoor settings for reflecting sunlight [23, 32] and indoors for establishing non-LoS 60Ghz wireless communication in data center networks [40].

[5]Leaves that can be reached by multiple masters, thus can be reassigned to different leaf sets based on a given optimization.

While energy distribution over multiple hops *(2)* and routing using special nodes *(3)* are interesting challenges, these topics require separate treatment as they are significantly beyond the scope of what we can evaluate with our existing deployment and simulation framework. In the following, we thus limit our focus on providing an exemplary solution to one particular functionality of the network layer, i.e., *(1)*, to update the leaf sets of the master nodes based on a global energy view of the network.

6.2 The Networking Problem

Similarly to the scheduling layer, where we have multiple choices of algorithms for energy distribution, the leaf set allocation can also be optimized differently based on the given application requirements. This ranges from simple optimizations, such as balancing the energy deficit across all masters, to prioritizing certain masters because of the importance of their network section (or leaf set) from the application's view point.

At the network layer, masters periodically exchange the *energy-state* information with each other. The energy-state may include the energy deficit d_m of a master m, the current leaf set ls_m along with individual energy demands of each leaf, and the set R_{ls_m} of leaves reachable but currently not necessarily in ls_m. The energy deficit can either be positive (supply exceeds demand) or negative (demand exceeds supply). The aforementioned information allows to create a global energy view of the network that, for example, can be used to dynamically adjust leaf set allocation to create an *energy balance* in the network. We define this energy balancing problem as a leaf set allocation that reduces the gap between the maximum and minimum energy deficit among masters in the network below a certain threshold τ:

$$d_{m_{max}} - d_{m_{min}} < \tau. \tag{6}$$

We do not restrict the choice of algorithm to solve this problem at the network layer; it can either be centralized or distributed. However, for our initial implementation we prefer a centralized approach where masters can elect a leader using any of the well-known leader selection protocols [11] to run the following leaf set allocation algorithm.

6.3 Algorithm

Our proposed network layer algorithm (cf. Algorithm 4) tries to iteratively reduce the energy deficit gaps among masters below τ. The idea here is to detect outliers from deficit observations and reallocate leaf set assignments to eventually eliminate these outliers. The algorithm starts by calculating the median deficit. For each outlying master m, i.e., the masters with higher deficits (i.e., $|median - d_m| > \tau/2$), it tries to find another master with lowest (or positive) deficit and overlapping reachable leaves. It then readjusts the leaf set of both the masters. Temporary new outliers can be accepted to dilute the deficit away from a corner in the network. If the algorithm does not terminate, then a balance might not be possible and a higher value for τ may have to be chosen. τ can initially be assigned a larger value which is iteratively reduced until no further improvements are possible. In a network with dynamic workloads and higher node churn, this energy balancing algorithm will have to be rerun frequently.

6.4 Evaluating Network Layer

In our evaluation, we want to see how this algorithm reacts (i) when a global energy view of the network is created for the first time and (ii) at the occurrence of node churn. We simulate a binary tree topology of 30 leaves charged by five masters using the DSF scheduling algorithm. Initially, with *energy state* of the network unknown, leaves are randomly distributed among masters. The PRR for this uniform distribution of leaves is depicted in Figure 16 for the first 50s of the simulation

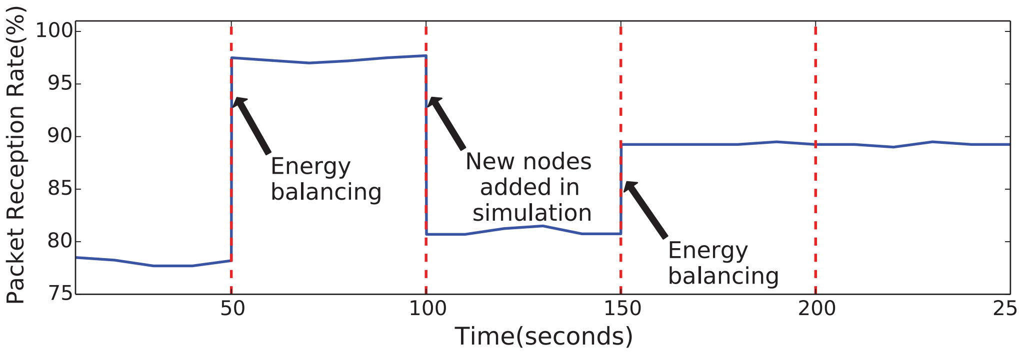


Fig. 16. Network layer: A global energy view of the network enabled by periodic information sharing between *masters* improves network wide performance through leaf set reassignments.

ALGORITHM 4: Network Wide Energy Balancing

Data: $LS := \{ls_{m_i} : m_i \in \mathbb{M}_a\}, MD := \{d_{m_i} : m_i \in \mathbb{M}_a\}, \{t_{l_i} : l_i \in ls_m\}, R_{ls_m}, \tau$ // τ:deficit threshold

Result: $\{ls_{m_i} : m_i \in \mathbb{M}_a\}$ // updated leafset for all masters in $\mathbb{M}_a$

while $|\text{median } MD - \max MD| > \frac{\tau}{2}$ **do**

  $o \leftarrow \text{argmax}_{m_i} MD$ // o is the master with the highest deficit, // the outlier

  $c \leftarrow \text{argmin}_{m_i} \{d_{m_i} : ls_o \cap R_{ls_{m_i}}\}$ // c is the master with the lowest deficit // with a leaf currently assigned to o that is reachable by c

  $l \leftarrow \text{argmax}_{l_i} \{t_{l_i} : l_i \in ls_o \cap R_{ls_c}\}$ // get the heaviest leaf // assigned to o that is reachable by c

  $ls_c \leftarrow ls_c \cup \{l\}$ // reassign leaf to c

  $ls_o \leftarrow ls_o \setminus \{l\}$ // and remove from o

  $d_c \leftarrow d_c + t_l$ // update deficits

  $d_o \leftarrow d_o - t_l$

end

run. However, at this point in time, masters exchange energy-state information with other. This results in dynamic update of the leaf set assignment with the aim to minimize network wide energy deficit, thereby improving the PRR from 78% to 98%. At 100s, we now add four more nodes into the simulation randomly assigned to masters. We again see a sudden decrease in PRR as the leaf set assignment is yet not optimized to minimize the network wide energy deficit. With the next round of energy-state exchange at 150s and the resulting leaf set updates, we can see improvement in PRR from that point in time onwards. Overall, this simple experiment magnifies the need for a network layer abstraction in the energy stack to optimize network wide energy distribution and performance.

7 RELATED WORK

The scope of this work is quite novel: We are unaware of any other proposal on a layered approach to energy management. The key contribution is a complete software and network architecture providing the abstraction of a decoupled energy plane. We can broadly divide the related research in three main categories.

Energy Neutral Operation. The work in this category tries to optimize a node's local behavior to achieve energy neutral operation in battery-less networks [8, 31, 39]. For example, TULA [31] balances the sensing and communication rates of a node based on its harvested energy, as sensing more data than what can be delivered is not useful. Similarly, TULA ensures that all the downstream neighbors do not forward more packets than a node can handle to achieve network wide energy neutral operation. Buchli et al. [8] present an approach to achieve uninterrupted WSN operation over time periods on the order of multiple years. The authors first study the annual dynamics of an energy source to develop an appropriate power subsystem before deployment and then dynamically compute the long-term sustainable performance level at runtime. Both the approaches are orthogonal as well as complementary to our work. We strongly believe that, as a future work, a network wide energy distribution architecture can be coupled with approaches that optimize a node's behavior locally to further enhance the performance of battery free networks.

Energy Transfer. Recent literature evaluates both wireless [13, 22, 23, 32, 35, 37], such as LEDs [23] or radio transmissions [9, 27], and wired energy transfer mechanisms [41, 42] for energizing WSN. All these efforts are particularly focused on establishing the feasibility and evaluating the practical limits of the corresponding energy transfer mechanism. We believe that these efforts provide us with useful hints whether a certain technology can be employed at the transfer layer of the energy stack. Alternatively, our proposed architecture provides a basic setup where these approaches can be plugged in at the transfer layer in a modular way and reap the benefits of a layered approach to network wide energy management.

Mobile Charging. Another proposal for energy transfer in WSN is to physically carry energy through the network [16, 22, 25] and recharge node batteries. The research in this area mainly focuses on an optimization problem of finding the best path through the network that results in recharging the maximum number of nodes in minimal time. While we also propose limited mobility support for master nodes to overcome line-of-sight challenges, the proposal for network wide recharging through a mobile robot has yet to achieve practical significance. This is because the requirement of a human-driven or robotic vehicular movement across the network strongly limits deployment scenarios.

8 CONCLUSIONS AND FUTURE WORK

We presented a new software and network model for enabling perpetual network deployments based on WET. The software model proposes a separate energy stack, with an initial specification of three layers, to efficiently manage energy distribution in the network. Our evaluation, using both experimental deployments and high level simulations, establishes the feasibility of a layered solution to energy management.

We are currently working on fine tuning the operation of the three layers of the energy stack. For example, at the transfer layer we are exploring the possibility of employing different uni- and omni-directional WET mechanisms, both individually and simultaneously, and addressing related challenges for the transfer protocol and defining related API. Similarly, we want to further evaluate the proposed scheduling algorithms in more application specific scenarios and with specialized performance parameters. Nonetheless, the network layer forms the bulk of our future work as it presents us with completely new challenges related to energy routing. Besides developing new algorithms, we are trying to understand the performance of the proposed energy balancing algorithm by varying the degree of leaf set overlap between multiple master nodes. Similarly, network layer challenges are different when using omnidirectional WET. For example, as opposed to the reshuffling leaf sets, the energy transfer by multiple masters needs to be interleaved taking into account the energy availability patterns as well as the degree of leaf overlapping between multiple masters.

REFERENCES

[1] Muhammad Hamad Alizai, Qasim Raza, Yasra Chandio, Affan A. Syed, and Tariq M. Jadoon. 2016. Simulating intermittently powered embedded networks. In *Proceedings of the 2016 International Conference on Embedded Wireless Systems and Networks (EWSN'16)*. Junction Publishing, 35–40.

[2] Abdalkarim Awad, Thorsten Frunzke, and Falko Dressler. 2007. Adaptive distance estimation and localization in wsn using RSSI measures. In *Proceedings of the 10th Euromicro Conference on Digital System Design Architectures, Methods and Tools (DSD'07)*. IEEE Computer Society, 471–478. DOI: http://dx.doi.org/10.1109/DSD.2007.20

[3] Guillermo Barrenetxea, François Ingelrest, Gunnar Schaefer, and Martin Vetterli. 2008. The hitchhiker's guide to successful wireless sensor network deployments. In *Proceedings of the 6th ACM Conference on Embedded Network Sensor Systems (SenSys'08)*. ACM, New York, NY, 43–56. DOI: http://dx.doi.org/10.1145/1460412.1460418

[4] Naveed Anwar Bhatti, Muhammad Hamad Alizai, Affan A. Syed, and Luca Mottola. 2016. Energy harvesting and wireless transfer in sensor network applications: Concepts and experiences. *ACM Trans. Sen. Netw.* 12, 3, Article 24 (Aug. 2016), 40 pages. DOI: http://dx.doi.org/10.1145/2915918

[5] Naveed Anwar Bhatti and Luca Mottola. 2017. HarvOS: Efficient code instrumentation for transiently-powered embedded sensing. In *Proceedings of the 16th ACM/IEEE International Conference on Information Processing in Sensor Networks (IPSN'17)*. ACM, New York, NY, 209–219. DOI: http://dx.doi.org/10.1145/3055031.3055082

[6] Naveed Anwar Bhatti, Affan Ahmed Syed, and Muhammad Hamad Alizai. 2014. Sensors with lasers: Building a WSN power grid. In *Proceedings of the 13th International Symposium on Information Processing in Sensor Networks (IPSN'14)*. IEEE Press, Piscataway, NJ, 261–272.

[7] Edoardo S. Biagioni and K. W. Bridges. 2002. The application of remote sensor technology to assist the recovery of rare and endangered species. *Int. J. High Perf. Comput. Appl.* 16, 3 (2002), 315–324.

[8] Bernhard Buchli, Felix Sutton, Jan Beutel, and Lothar Thiele. 2014. Dynamic power management for long-term energy neutral operation of solar energy harvesting systems. In *Proceedings of the 12th ACM Conference on Embedded Network Sensor Systems (SenSys'14)*. ACM, New York, NY, 31–45. DOI: http://dx.doi.org/10.1145/2668332.2668333

[9] Michael Buettner, Richa Prasad, Alanson Sample, Daniel Yeager, Ben Greenstein, Joshua R. Smith, and David Wetherall. 2008. RFID sensor networks with the intel WISP. In *Proceedings of the 6th ACM Conference on Embedded Network Sensor Systems (SenSys'08)*. ACM, New York, NY, 393–394. DOI: http://dx.doi.org/10.1145/1460412.1460468

[10] M. Ceriotti, M. Corrà, L. D'Orazio, R. Doriguzzi, D. Facchin, S. T. Gună, G. P. Jesi, R. L. Cigno, L. Mottola, A. L. Murphy, M. Pescalli, G. P. Picco, D. Pregnolato, and C. Torghele. 2011. Is there light at the ends of the tunnel? Wireless sensor networks for adaptive lighting in road tunnels. In *Proceedings of the 2011 10th International Conference on Information Processing in Sensor Networks (IPSN'11)*. 187–198.

[11] George Coulouris, Jean Dollimore, Tim Kindberg, and Gordon Blair. 2011. *Distributed Systems: Concepts and Design (5th ed.)*. Addison-Wesley.

[12] CreeQ3. 2016. 3W CREE Q3 LED Flashlight. Retrieved from http://tinyurl.com/3WFlashlight.

[13] R. Doost, K. R. Chowdhury, and M. Di Felice. 2010. Routing and link layer protocol design for sensor networks with wireless energy transfer. In *Proceedings of the IEEE Global Telecommunications Conference (GLOBECOM'10)*. 1–5. DOI: http://dx.doi.org/10.1109/GLOCOM.2010.5683334

[14] Deborah Estrin and others. 1999. Next century challenges: Scalable coordination in sensor networks. In *Proceedings of the 5th Annual ACM/IEEE International Conference on Mobile Computing and Networking (MobiCom'99).*

[15] Omprakash Gnawali, Ben Greenstein, Ki-Young Jang, August Joki, Jeongyeup Paek, Marcos Vieira, Deborah Estrin, Ramesh Govindan, and Eddie Kohler. 2006. The tenet architecture for tiered sensor networks. In *Proceedings of the 4thACM SenSys Conference*. ACM, 153–165.

[16] Liang He, Lingkun Fu, Likun Zheng, Yu Gu, Peng Cheng, Jiming Chen, and Jianping Pan. 2014. ESync: An energy synchronized charging protocol for rechargeable wireless sensor networks. In *Proceedings of the 15th ACM International Symposium on Mobile Ad Hoc Networking and Computing (MobiHoc'14)*. ACM, New York, NY, 247–256. DOI: http://dx.doi.org/10.1145/2632951.2632970

[17] Tian He, Chengdu Huang, Brian M. Blum, John A. Stankovic, and Tarek Abdelzaher. 2003. Range-free localization schemes for large scale sensor networks. In *Proceedings of the 9th Annual International Conference on Mobile Computing and Networking (MobiCom'03)*. ACM, New York, NY, 81–95. DOI: http://dx.doi.org/10.1145/938985.938995

[18] IXYS. 2007. SolarMD SLMD481H08L. Retrieved from http://ixapps.ixys.com.

[19] Bryce Kellogg, Vamsi Talla, Shyamnath Gollakota, and Joshua R. Smith. 2016. Passive wi-fi: Bringing low power to wi-fi transmissions. In *Proceedings of the 13th USENIX Symposium on Networked Systems Design and Implementation (NSDI 16)*. USENIX Association, Santa Clara, CA, 151–164.

[20] Loukas Lazos and Radha Poovendran. 2004. SeRLoc: Secure range-independent localization for wireless sensor networks. In *Proceedings of the 3rd ACM Workshop on Wireless Security (WiSe'04)*. ACM, New York, NY, 21–30. DOI: http://dx.doi.org/10.1145/1023646.1023650

[21] Philip Levis, Nelson Lee, Matt Welsh, and David Culler. 2003. TOSSIM: Accurate and scalable simulation of entire TinyOS applications. In *Proceedings of the 1st International Conference on Embedded Networked Sensor Systems (SenSys'03)*. ACM, New York, NY, 126–137. DOI: http://dx.doi.org/10.1145/958491.958506

[22] K. Li, H. Luan, and C. C. Shen. 2012. Qi-ferry: Energy-constrained wireless charging in wireless sensor networks. In *Proceedings of the 2012 IEEE Wireless Communications and Networking Conference (WCNC'12)*. 2515–2520. DOI: http://dx.doi.org/10.1109/WCNC.2012.6214221

[23] Peng Liu, Yifan Wu, Jian Qiu, Guojun Dai, and Tingting Fu. 2013. eLighthouse: Enhance solar power coverage in renewable sensor networks. *Int. J. Distrib. Sens. Netw.* 9, 11 (2013). DOI: http://dx.doi.org/10.1155/2013/256569

[24] Andrea Massa, Giacomo Oliveri, Federico Viani, and Paolo Rocca. 2013. Array designs for long-distance wireless power transmission: State-of-the-art and innovative solutions. *Proc. IEEE* 101, 6 (2013), 1464–1481. DOI: http://dx.doi.org/10.1109/JPROC.2013.2245491

[25] Yang Peng, Zi Li, Wensheng Zhang, and Daji Qiao. 2010. Prolonging sensor network lifetime through wireless charging. In *Proceedings of the 31st IEEE Real-Time Systems Symposium (RTSS'10)*. 129–139. DOI: http://dx.doi.org/10.1109/RTSS.2010.35

[26] Enrico Perla, Art Ó. Catháin, Ricardo Simon Carbajo, Meriel Huggard, and Ciarán Mc Goldrick. 2008. PowerTOSSIM Z: Realistic energy modelling for wireless sensor network environments. In *Proceedings of the 3nd ACM Workshop on Performance Monitoring and Measurement of Heterogeneous Wireless and Wired Networks (PM2HW2N'08)*. ACM, New York, NY, 35–42. DOI: http://dx.doi.org/10.1145/1454630.1454636

[27] PowerCast. 2009. P2110 receiver and TX91501-3W-ID transmitter. Retrieved from http://tinyurl.com/powercaster.

[28] Benjamin Ransford, Jacob Sorber, and Kevin Fu. 2011. Mementos: System support for long-running computation on RFID-scale devices. *SIGARCH Comput. Archit. News* 39 (2011), 159–170.

[29] Thomas Schmid, Roy Shea, Mani B. Srivastava, and Prabal Dutta. 2010. Disentangling wireless sensing from mesh networking. In *Proceedings of the 6th Workshop on Hot Topics in Embedded Networked Sensors (HotEmNets'10)*. ACM, New York, NY. DOI: http://dx.doi.org/10.1145/1978642.1978646

[30] Fabio Silva and others. 2009. SPAN: A sensor processing and acquisition network—Field deployment lessons learned. In *Proceedings of the 94th ESA Annual Meeting*.

[31] J. Sorber, A. Balasubramanian, M. D. Corner, J. R. Ennen, and C. Qualls. 2013. Tula: Balancing energy for sensing and communication in a perpetual mobile system. *IEEE Transa.ctions on Mobile Comput.* 12, 4 (2013), 804–816. DOI: http://dx.doi.org/10.1109/TMC.2012.52

[32] Affan A. Syed, Young Cho, and John Heidemann. 2010. Energy transference for sensornets. In *Proceedings of the 8th ACM Conference on Embedded Networked Sensor Systems (SenSys'10)*. ACM, New York, NY, 397–398. DOI: http://dx.doi.org/10.1145/1869983.1870041

[33] Nikola Tesla. 1919. *My Inventions: The Autobiography of Nikola Tesla*. Experimenter Publishing Company, Inc., New York, NY.

[34] Ning Wang, Yong Zhu, Wei Wei, Jianjun Chen, Shenshen Liu, Ping Li, and Yumei Wen. 2012. One-to-multipoint laser remote power supply system for wireless sensor networks. *IEEE Sensors Journal* 12 (2012), 389–396.

[35] Zhihua Wang, Orfeas Kypris, and Andrew Markham. 2016. RePWR: Wireless power transfer within reinforced concrete. In *Proceedings of the 4th International Workshop on Energy Harvesting and Energy-Neutral Sensing Systems (ENSsys'16)*. ACM, New York, NY, 1–6. DOI: http://dx.doi.org/10.1145/2996884.2996885

[36] Wuhan Lilly Electronics Co.2016. Focusable 0.8W near-infrared IR laser diode module. Retrieved from http://tinyurl.com/0-8WLaser.

[37] L. Xie, Y. Shi, Y. T. Hou, and H. D. Sherali. 2012. Making sensor networks immortal: An energy-renewal approach with wireless power transfer. *IEEE/ACM Trans. Netw.* 20, 6 (Dec. 2012), 1748–1761. DOI: http://dx.doi.org/10.1109/TNET.2012.2185831

[38] Xiaoyu Zhang, Hanjun Jiang, Lingwei Zhang, Chun Zhang, Zhihua Wang, and Xinkai Chen. 2010. An energy-efficient ASIC for wireless body sensor networks in medical applications. *IEEE Trans. Biomed. Circuits and Systems* 4, 1 (2010), 11–18.

[39] Yuting Zhang, Thomas D. C. Little, Benjamen R. Wetherill, Francesco Peri, and Robert F. Chen. 2014. An instrument scheduler design for energy neutral coastal monitoring systems deployment. In *Proceedings of the 2nd International Workshop on Energy Neutral Sensing Systems (ENSsys'14)*. ACM, New York, NY, 1–6. DOI: http://dx.doi.org/10.1145/2675683.2675684

[40] Xia Zhou, Zengbin Zhang, Yibo Zhu, Yubo Li, Saipriya Kumar, Amin Vahdat, Ben Y. Zhao, and Haitao Zheng. 2012. Mirror mirror on the ceiling: Flexible wireless links for data centers. In *Proceedings of the ACM SIGCOMM 2012 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communication (SIGCOMM'12)*. ACM, New York, NY, 443–454. DOI: http://dx.doi.org/10.1145/2342356.2342440

[41] Ting Zhu, Yu Gu, Tian He, and Zhi-Li Zhang. 2010. eShare: A capacitor-driven energy storage and sharing network for long-term operation. In *Proceedings of the 8th ACM Conference on Embedded Networked Sensor Systems (SenSys'10)*. ACM, New York, NY, 239–252. DOI: http://dx.doi.org/10.1145/1869983.1870007

[42] Ting Zhu, Yu Gu, Tian He, and Zhi-Li Zhang. 2012. Achieving long-term operation with a capacitor-driven energy storage and sharing network. *ACM Trans. Sen. Netw.* 8, 4, Article 32 (Sept. 2012), 37 pages. DOI: http://dx.doi.org/10.1145/2240116.2240121

Received January 2017; revised January 2018; accepted February 2018