Poster: HarvNet: Battery-Free Device Network Simulator

Hrishikesh G. Kusneniwar
Department of CS&IS,
BITS Pilani, Goa Campus, India

Sougata Sen
Department of CS&IS, APPCAIR,
BITS Pilani, Goa Campus, India

Josiah D. Hester
College of Computing,
Georgia Institute of Technology, USA

ABSTRACT

Ubiquitous sensing technologies, leveraging a network of interconnected sensors and devices, offer multifaceted benefits to society. However, the use of batteries has persistently posed a challenge to their advancement. In recent years, researchers have explored establishing communication between battery-free nodes. The primary objective of this work is to expedite the testing of algorithms for multiple battery-free interconnected nodes by isolating the algorithm from the underlying hardware. Isolating the hardware allows faster tuning of algorithms, and enables testing on a large scale. We developed a novel python-based simulation framework, *HarvNet*. Simulations using real-world power traces demonstrated a success rate of 81% in establishing connections between nodes which closely emulates the success rate achieved using hardware.

CCS CONCEPTS

• **Human-centered computing** → **Ubiquitous computing**; • **Networks** → *Network simulations.*

KEYWORDS

Ubiquitous computing, Battery-Free device networks

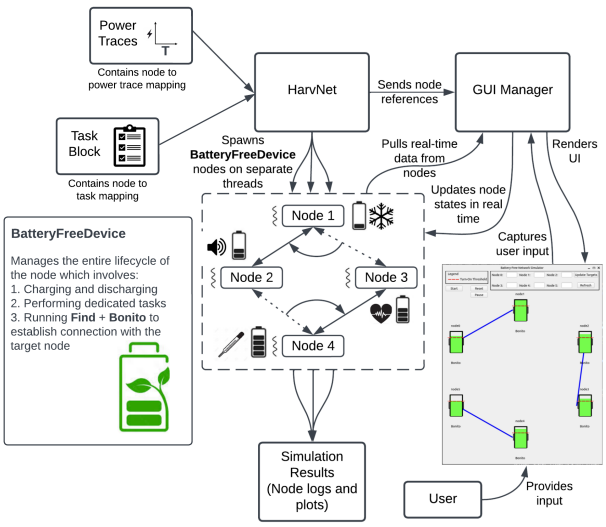
1 INTRODUCTION

Utilizing a network of interconnected sensors and devices provides a wide array of benefits in multiple domains [5]. This has led the proliferation of such inter-connected devices. But the reliance of these devices on batteries has consistently posed a deployment challenge. The innovation in the area of battery-free systems has enabled cheap and lightweight devices to perform the same tasks using harvested energy [4]. These devices commonly have inter-device communication requirements, and novel network communication algorithms are being developed for them. However, such algorithms often have hardware dependence and testing them on devices is a challenge. Thus needed is a framework that enables testing network communication for intermittently powered battery-free IoT nodes.

A battery-free device cannot avoid power failures, giving rise to its intermittent nature of operation. To be able to communicate, a sender and a receiver must be active and have enough energy for at least one complete packet transmission at the same time. Since the nodes' activity phases are generally interleaved and short, it often takes thousands of wake-ups until two nodes encounter each other and communication becomes possible. *Find* is a neighbor discovery protocol for battery-free wireless devices that uses randomized waiting to minimize discovery latency [2]. *Bonito* is another inter-device communication protocol that learns the the charging time distribution and develops a connection protocol that enables reliable and efficient bi-directional communication between intermittently powered nodes [1]. However, Bonito only achieves communication between a pair of devices. Various trade-offs and challenges emerge from each potential approach to transitioning from a two-node scenario to larger networks, warranting exploration in the field [1].

To address this gap, we developed HarvNet, a python-based simulation framework that currently leverages the Bonito protocol to enable research of coordinating tasks across a network of intermittent nodes. In this paper, we describe the design of HarvNet, and approaches that we undertook to validate the system. The **key contributions** of this work are as follows. First, we developed a simulation framework (HarvNet) that enables coordination of tasks in a network of intermittent nodes. Second, we translated the entire Bonito protocol to python validated using simulations on real world power traces that provides a convenient setup to swiftly gauge the impact of tuning parameters or making subtle modifications. Although we currently tested *Bonito* with *HarvNet*, however HarvNet can easily enable deployment and testing of other protocols.

Figure 1: HarvNet System Design



2 SYSTEM DESIGN OF HARVNET

Figure 1 presents the system design of HarvNet. HarvNet consists of multiple nodes, each of which aims to communicate with another node in the network for any application related tasks. Each HarvNet node is an intermittently powered device that is aware of the existence of other nodes in the network. However, a node has

MOBISYS '24, June 3–7, 2024, Minato-ku, Tokyo, Japan

ACM ISBN 979-8-4007-0581-6/24/06…$15.00
https://doi.org/10.1145/3643832.3661444

Table 1: Actual Data vs HarvNet simulation

Node	Total no. of wakeups	Successful Connections	Success Rate
nRF52805MCU	63,072	61,868	99.89%
HarvNet BFD	63,995	51,751	80.87%

no prior information about other node's energy levels or their wake up cycles. One must note that every node can wake up at different times and they do not have the same energy capacity. Each HarvNet node has the following functionalities implemented. (1) **A utility** that manages the entire lifecycle of the node, involving charging and discharging itself, and running the desired communication protocol. This utility currently manages the Bonito protocol. (2) **A dedicated callback** that executes every time the node wakes up after recharging. This callback is responsible for running the task allocated to a node. Users can conveniently define a separate task for each node in the HarvNet's task script. (3) **Bonito protocol** implementation which will allow a node to form and maintain connections with a target node in the network.

HarvNet has a custom GUI displaying real-time node energy levels and active connections to aid the visual understanding of communication in intermittent devices. The number of nodes and their corresponding power traces are predetermined using the input file. The main simulation environment then instantiates these node as a **BatteryFreeDevice** that simulates a battery free node in a distributed setup. Once the simulation starts, HarvNet launches all these virtual BatteryFreeDevice nodes to work independently of other nodes in the simulation environment. Nodes are now independently charging and discharging, forming connections and performing the dedicated tasks. The initial target of each node for establishing connections is specified through the GUI by clicking on specific nodes, and this pairing can be updated in real-time. Although we currently implemented the Bonito protocol for communication, however HarvNet allows implementing a custom logic for the orchestration of connections among the nodes.

The connection formation among nodes is simulated using inter-thread communication. The nodes discover each other by accessing a specific state maintained by every node indicating whether it is awake or asleep. A node's utility sets this variable to high as soon as the node wakes up. This nodes waits for a certain period of time (In the current Bonito implementation, two nodes can successfully detect each other if they wake up with an offset between 88 μs and 848 μs.) to establish a connection with other nodes. The confirmation of a successful connection is simulated by making use of a **Barrier**, which is a synchronization primitive in Python. It allows a node to wait on the same barrier object instance for a certain duration (i.e. at the same point in code), until the target node arrives.

3 RESULTS

We used the power traces from **Stairs Dataset** [3] to simulate Bonito protocol on HarvNet to validate it against the actual tests performed by the authors of Bonito [1]. The results presented in Table 1 demonstrate that HarvNet closely emulates the wake ups in the real world. However, the successful connection rate in HarvNet is lower. We identified that the thread safe event variable used to mimic the process of establishing connection between two nodes has time overhead for acquiring locks to avoid race conditions results, which results in multiple near misses.

4 APPLICATIONS AND CONCLUSION

Data Ferrying: The ability to transfer data from one from one distant node to another, not in transmission range of each other (data ferrying), would be essential. HarvNet can be conveniently used to test solutions for the above problem. An example strategy that could be tested on HarvNet involves employing the Breadth-First Search (BFS) algorithm on this graph of intermittent devices. Every node has some amount of non-volatile memory to store the graph and also has the BFS algorithm coded into the task block. To start transporting the data, the novel communication protocol will be invoked to establish a connection with the next target node in the path. Data (including the path) will be transferred in chunks over several encounters (due to the energy shortage). The target node then propagates the data in a similar way. We can easily scale the machine running HarvNet to simulate a network of thousands of nodes, demonstrating its scalability.

Network Awareness: Network awareness is the problem of notifying all the nodes in a network about an event that has occurred. An intermittent node will not be able to broadcast packets for enough time for all nodes to be able to receive them. This intermittent nature will also reduce the listening window of the receiver nodes, further reducing the chances of receiving the packet. A possible strategy that could be tested on HarvNet involves sharing information by establishing connections instead of broadcasting packets. On wake-up, a node can attempt to establish connection with any active node that is not notified yet. It also maintains a list of all nodes that have already received the information. Now, it updates the list and shares both the information and the list, so that the target node can also propagate the data. In this way, a butterfly effect is achieved.

Conclusion: The present work provides a python-based simulation framework that takes a real-world power trace as an input and allows the user to run simulations of Battery-free nodes enabling tweaks during runtime. This framework tries to imitate the unpredictability and randomness of the real world. This isolation of hardware allows for quicker iterations of tuning of the algorithm and testing on a large scale.

ACKNOWLEDGEMENT

This work is partly sponsored by BITS Pilani's grant # C1/23/173. All findings and recommendations are those of the authors and do not necessarily reflect the views of the funding institute.

REFERENCES

[1] Kai Geissdoefer and Marco Zimmerling. 2022. Learning to Communicate Effectively Between Battery-free Devices. In *USENIX NSDI*.
[2] Kai Geissdoerfer and Marco Zimmerling. 2021. Bootstrapping Battery-free Wireless Networks: Efficient Neighbor Discovery and Synchronization in the Face of Intermittency. In *USENIX NSDI*.
[3] Kai Geissdoerfer and Marco Zimmerling. 2022. Time-synchronized energy harvesting traces. In *[Data set] USENIX NSDI*. https://doi.org/10.5281/zenodo.6383042
[4] Josiah Hester and Jacob Sorber. 2017. The Future of Sensing is Batteryless, Intermittent, and Awesome. In *ACM Sensys'17*.
[5] Dionisis Kandris, Christos Nakas, Dimitrios Vomvas, and Grigorios Koulouras. 2020. Applications of Wireless Sensor Networks: An Up-to-Date Survey. *Applied System Innovation* (2020).