A Little Certainty is All We Need: Discovery and Synchronization Acceleration in Battery-Free IoT

Gaosheng Liu*
Vrije Universiteit Amsterdam

Vinod Nigade*
Vrije Universiteit Amsterdam

Henri Bal
Vrije Universiteit Amsterdam

Lin Wang
Paderborn University

ABSTRACT

The vision of sustainable IoT constructed from battery-free devices has attracted ample interest in the research community. Yet, efficient device discovery and synchronization—a fundamental problem in IoT systems—remains a critical challenge mainly due to the uncertain ambient energy availability across battery-free devices. We argue that bringing in a small level of certainty is necessary for facilitating communication in battery-free IoT. We propose Pulsar where we introduce a small number of battery-powered devices, serving as the communication coordinator for a large number of battery-free devices. We develop two communication schemes, namely one-to-one, and all-to-all, for Pulsar. Our results based on simulations and prototype-based experiments show that Pulsar achieves consistently good performance across different scenarios while requiring no special hardware or environmental conditions.

1 INTRODUCTION

Recently, battery-free (BF) embedded devices have gained tremendous attention due to their promise of delivering a more sustainable Internet-of-Things (IoT) ecosystem [1, 4, 5, 12, 20]. Traditionally, IoT systems employ embedded devices powered by batteries that contain hazardous chemicals. Moreover, the onboard battery, when depleted or broken, must be recharged/replaced to sustain the system. However, IoT systems are typically deployed in a geographically dispersed fashion (e.g., precision agriculture in a large farm [9, 21] or even embedded into physical infrastructures (e.g., construction health monitoring [15]), making the maintenance work extremely challenging. BF embedded devices save us from this hassle by scavenging energy directly from the environment, thus eliminating the need for batteries and their associated maintenance costs. It is envisioned that a completely sustainable IoT system can be built solely based on BF devices.

While the vision is appealing, the reality is cruel. The main challenge arises from the uncertainty in energy availability. Since ambient energy is too limited to power the device continuously, BF devices typically employ a small capacitor to buffer the harvested energy and work intermittently and transiently (e.g., for several

*Both authors contributed equally to this research.

APNet 2024, August 3–4, 2024, Sydney, Australia
© 2024 Copyright held by the owner/author(s).
ACM ISBN 979-8-4007-1758-1/24/08.
https://doi.org/10.1145/3663408.3663414

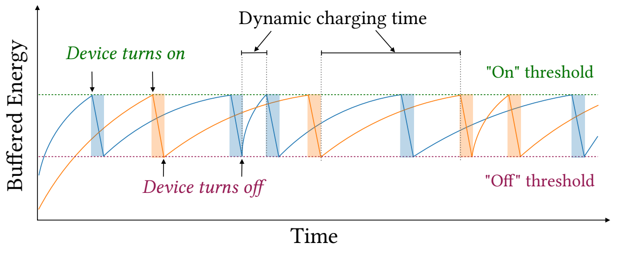


Figure 1: Intermittent working style of BF devices with dynamic charging times between device wake-ups [4].

milliseconds [2, 3, 19])—waking up to execute the program when the capacitor voltage reaches the "on" threshold and falling asleep to recharge the capacitor when the voltage drops below the "off" threshold, as depicted in Figure 1. To combat intermittency and ensure the forward progress of program execution, research has been focused on programming models and compilers for a single BF device [11, 16]. Yet, the communication between multiple BF devices, an essential functionality of an IoT system, has been underexplored.

One critical issue in communication between BF devices is the discovery of other devices and the synchronization of their wake-ups. To enable communication, two battery-free, intermittently-working devices must be active simultaneously. This trivial condition in continuously powered systems becomes hard since the harvested energy by each device varies over time and can be highly unpredictable (see Figure 2) [2, 4, 8, 17]. Existing works have attempted this issue typically under the assumption of homogeneous and/or static charging times of communicating devices [4, 13, 14]. One of the key ideas, proposed in Find [4], is based on postponing the wake-ups of (one of) the devices randomly following a fine-tuned distribution. While being effective under homogeneous conditions, this approach is susceptible to environmental changes and is not efficient in cases with heterogeneous and dynamic charging times. In fact, with the increase of charging time variability, the performance of such an approach may deteriorate to that of a passive greedy approach where BF devices wake up immediately when charged without any delay (see Figure 7).

We argue that to facilitate the discovery and communication of BF devices under unpredictable charging time variability, it is necessary to introduce some degree of certainty in the system. This certainty serves as shared knowledge among BF devices so they have some rough expectations concerning the behavior of other devices. Our insight is that such certainty can be achieved by the introduction of one or a small number of battery-powered devices among multiple BF devices.

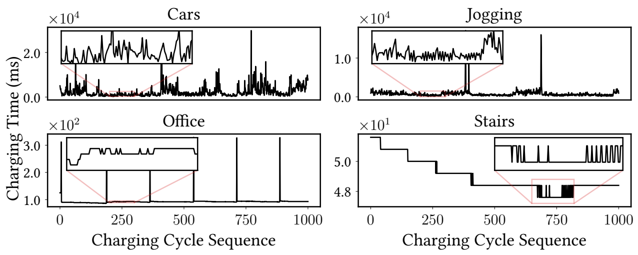
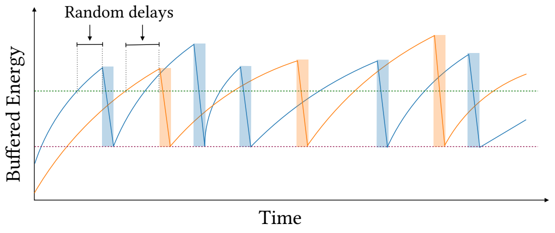
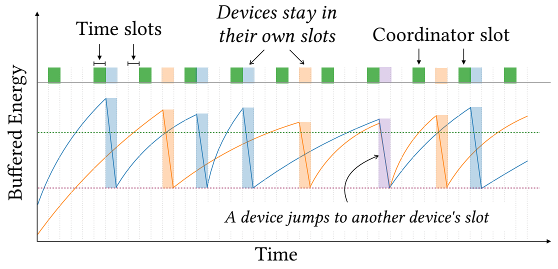


Figure 2: Charging time traces of BF devices.



(a) Random delay based approach adopted by Find [4] where BF devices postpone their wake-ups with a random delay following a fine-tuned distribution.


Time slots
Devices stay in their own slots
Coordinator slot
Buffered Energy
A device jumps to another device's slot
Time

(b) High-level idea of Pulsar where BF devices are synchronized and coordinated by a battery-powered duty-cycled coordinator with scheduled communication plans.

Figure 3: Random delay based approaches vs. Pulsar.

We propose Pulsar, a system for accelerated discovery and synchronization among BF devices leveraging the minimum help from duty-cycled battery-powered devices (as coordinators). BF devices in Pulsar first discover the coordinator and then follow a scheduled communication scheme disseminated by the coordinator. The coordinator is duty-cycled with a fixed interval and can be easily discovered by a BF device. The coordinator also comes up with a scheduled communication scheme adaptively, where each BF device in the system is assigned a unique time slot, and such information is announced to all other BF devices in the system. Our preliminary evaluation based on simulations and prototype-based experiments shows that with the coordinator, the discovery and synchronization among BF devices can be constantly accelerated while requiring no special hardware or environmental conditions. We present the design of Pulsar and discuss its limitations as future work.

2 BACKGROUND AND MOTIVATION

2.1 Uncertainty in Battery-free IoT

IoT systems built from BF devices suffer from poor efficiency in computing and communication due to device intermittency. The root cause comes from the fact that the charging time of BF devices can be *highly variable and unpredictable* in many deployment scenarios due to the randomness in environmental conditions (e.g., the light condition on a solar panel and the signal strength for an RF energy harvester). This is in stark contrast to traditional duty-cycled battery-powered IoT where the device follows a pre-defined on-off schedule. To verify the charging time variability of BF devices, we pick snippets of 1000 charging cycles of a BF device randomly from charging time traces released in [5, 6]. Figure 2 depicts snippets for four different scenarios namely cars, jogging, office, and stairs, respectively. While some scenarios demonstrate higher dynamism than others, we observe high variability generally across most of these scenarios.

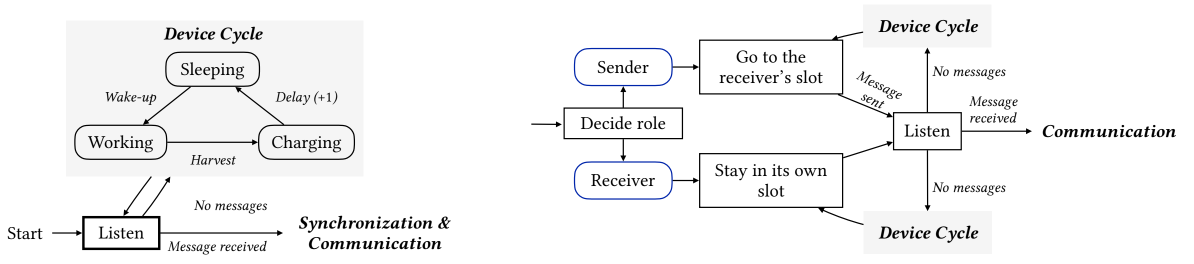
2.2 Motivation

Due to the unpredictable charging time of BF devices, the problem of device discovery and synchronization, a fundamental problem in communication in IoT systems, becomes challenging [4, 7, 18]. This is because, for two devices to discover and communicate, they must wake up simultaneously. However, due to the variability in charging time, it is hard for a BF device to know when exactly the target BF device will be sufficiently charged and turned on.

One popular approach to aligning the wake-ups of BF devices for communication is to postpone the wake-up of the BF device artificially with a delay, hoping to meet another device that happens to be on, as depicted in Figure 3(a). Geissdoerfer et al. show that randomly choosing the delay from a fine-tuned distribution can accelerate the discovery process in scenarios with homogeneous charging times [4]. However, the effectiveness drops in dynamic and heterogeneous charging environments, as shown in Figure 7. Bonito addresses, upon discovery, the synchronization problem between BF devices by exploiting statistical distributions of their charging times [5]. However, the device discovery at startup and after communication failure is based on Find.

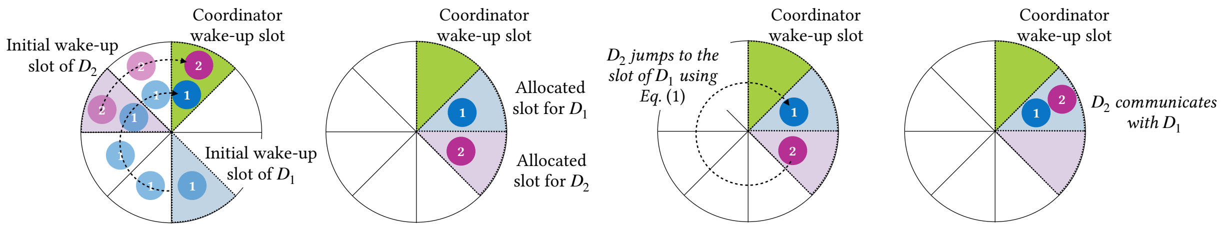
We argue that some degree of certainty is necessary in addressing the discovery and synchronization problem among BF devices efficiently. Our idea is to introduce a few battery-powered devices (with the same capabilities as the BF ones) as coordinators for BF devices. Exploiting the certainty exposed by the battery-powered device, BF devices can easily achieve time slot alignment and structured communication. As shown in Figure 3, the discovery and synchronization of two BF devices are facilitated by a coordinator, where BF devices are allocated specific communication slots shared among all devices in the systems. This allows BF devices to discover and synchronize with each other following a global slot allocation. The coordinator can be duty-cycled to prolong its own life.

At a high level, our idea is inspired by Flync [4] but goes beyond it in two major aspects: (1) Flync improves Find by aligning communication boundaries to increase the chance of discovery in a random communication procedure, but it does not support structured communication (e.g., different communication patterns) among BF devices as we do in Pulsar. (2) Flync requires special hardware support and a common energy signal that may not exist in many deployment scenarios, while Pulsar only requires attaching a battery to one or a few of the BF devices. There are also other hardware-based proposals orthogonal to ours [8].



(a) Coordinator discovery (b) Synchronization and communication

Figure 4: An overview of the simplified Pulsar workflow.


Coordinator wake-up slot
Initial wake-up slot of D_2
Initial wake-up slot of D_1
Allocated slot for D_1
Allocated slot for D_2
D_2 jumps to the slot of D_1 using Eq. (1)
D_2 communicates with D_1

(a) Coordinator discovery (b) Slot allocation (c) Slot jump for synchronization (d) Communication

Figure 5: Illustration of the coordinator discovery, slot allocation, and synchronization of BF devices.

3 PULSAR DESIGN

3.1 Overview

Pulsar consists of a number of BF devices and one or more battery-powered devices serving as coordinators. As a preliminary exploration, we assume one coordinator that operates on a time-invariant duty cycle. BF devices typically cycle through three successive states, beginning with the charging state allowing the device to accumulate sufficient energy before waking up to work. To start working in the desired slot (defined by the communication scheme), BF devices may enter the sleeping state [2, 4, 13] before performing the actual tasks like sensing, computing, and communicating in the working state. Given the fixed size of the capacitor and a typical sensing application, the duration of the working state is usually assumed constant [4]. The charging time (the duration of the charging state), on the other hand, can be variable and non-deterministic depending on the energy source conditions (see Figure 2).

Figure 4 sketches a simplified overview of the Pulsar workflow, depicting its two-stage process. In the first stage (Figure 4a), BF devices align their wake-ups with the active slot of the coordinator to establish the initial communication. To this end, a BF device systematically wakes up in every feasible slot of the coordinator's cycle by appropriately calculating the delay after being charged, as depicted in Figure 5(a). Once the alignment is achieved and communication with the coordinator is established, the BF device synchronizes the local time with the coordinator's time, precisely aligns its slot boundaries with those of the coordinator, and receives its designated working slot allocated by the coordinator; see Figure 5(b). In the second stage (Figure 4b), depending on the communication scheme, a BF device can function as either a sender or receiver. As a sender, the BF device jumps to the target receiver's slot to send (broadcast) messages by delaying its wake-up according to the communication scheme, in the hope that the receiver will also be active in its allocated slot and not in the charging phase. Conversely, when operating as a receiver, the BF device stays (wakes up) in its allocated slot to receive (broadcast) messages, as shown in Figure 5(c-d).

3.2 System Model

Pulsar, like many other wireless protocols [4, 10], divides time into equal-length slots. The length of these slots is usually defined according to the duration of the working state after a single wake-up, which in turn is determined by the size of the onboard capacitors. Typically, we can set the slot length in the range of 1 to 5 milliseconds for a 150 uF capacitor, supporting a few uninterrupted sensing and radio operations [2–4, 17]. The coordinator has the same slot length as BF devices. The duty cycle β of the coordinator determines the number of slots $T_{coor} = 1/\beta$ within its device cycle. The coordinator allocates specific slots to the BF devices from the *slot distribution cycle* of length T_{dist} where $T_{dist} = n \times T_{coor}$ for $n \in \mathbb{Z}^+$. The coordinator consistently wakes up at slot 0, marking the start of the slot distribution cycle. In this paper, we consider $n = 1$, meaning the coordinator's cycle and the slot distribution cycle have an equal number of slots. The value of T_{dist} should be at least equal to the maximum number of devices in the network to ensure every device has a unique designated slot.

Suppose the dynamic charging time for a BF device at device cycle k (i.e., k-th wake-up) is t_k, and the device working in slot i (in the slot distribution cycle) during the previous device cycle $k-1$ wants to stay in the same slot for device cycle k. After t_k slots in charging, the delay that needs to be applied at device cycle k to stay active in the same slot is calculated as $\delta_k = T_{dist} - (T_{dist} \bmod t_k) - 1$. To generalize, the delay d_k to jump from the previous slot i to any slot j in the slot distribution cycle is as follows.

$$d_k = \delta_k + \begin{cases} j - i & \text{if } j \geq i, \\ T_{dist} + (j - i) & \text{if } j < i. \end{cases} \quad (1)$$

The delay can be further optimized to avoid unnecessary jumps beyond T_{dist} as $d_k = (d_k \mod T_{dist})$ if $d_k > T_{dist}$. Hence, a BF device can efficiently wake up in any slot in the slot distribution cycle if it knows its charging time and previous working slot.

Pulsar relies on precise charging time measurements in each device cycle to correctly determine the delay to apply, thereby ensuring each device wakes up correctly at the desired slot. The precise measurement of charging times in current BF devices is feasible through the utilization of RTCs, which are also powered by ambient energy sources [2, 11, 16].

3.3 Two-Device Communication Scheme

Although BF devices can jump to desired slots efficiently, this feature alone does not ensure successful discovery or communication between devices. The challenge arises from the ambiguity in the roles of surrounding devices (whether they function as a sender or receiver) and the protocol for slot jumping (i.e., determining which device jumps to whose slot at what time). To address this challenge, we propose a simple yet effective communication scheme tailored for a two-device BF network.

As outlined in Section 3.1, BF devices initially discover and communicate with the coordinator by sequentially jumping through slots in the slot distribution cycle. To accomplish this sequential slot traversal, they set their wake-up delay as $d_k = \delta_k + 1$ during each device cycle k. The first BF device that successfully communicates with the coordinator, D_1, obtains the first free slot (i.e., slot 1) and assumes the role of receiver. Importantly, D_1 stays in its designated slot, following the protocol that specifies that all devices should stay in their designated slots and listen for incoming messages from other devices while functioning as receivers. Subsequently, the second BF device D_2 that obtains the next free slot (i.e., slot 2) is aware that slot 1 has already been allocated to another device. As a result, D_2 assumes the role of sender and jumps to slot 1 to establish communication with D_1. When both devices are active simultaneously in the same slot (i.e., slot 1), the communication will succeed eventually.

3.4 Multi-Device Communication Scheme

With multiple devices, finding an optimal schedule—dictating who communicates with whom and in what sequence—becomes exponentially hard due to the dynamic, heterogeneous, and unpredictable charging times of these devices. Overall, we must address two crucial challenges for optimal performance: *collision avoidance* to ensure that only one sender device communicates with the receiver device in a slot at any given time, and *role conflict resolution* to avoid both devices in a communication pair assume the same role (whether sender or receiver) to prevent potential deadlocks.

We propose a structured communication scheme to enable all-to-all communication among multiple BF devices. In this scheme, we assume that devices are within each other's communication range, including the coordinator, and are aware of the total number N of BF devices in the network. The communication protocol comprises phases and rounds as depicted in Figure 6. In each phase, devices construct sets of senders and receivers. Since every device

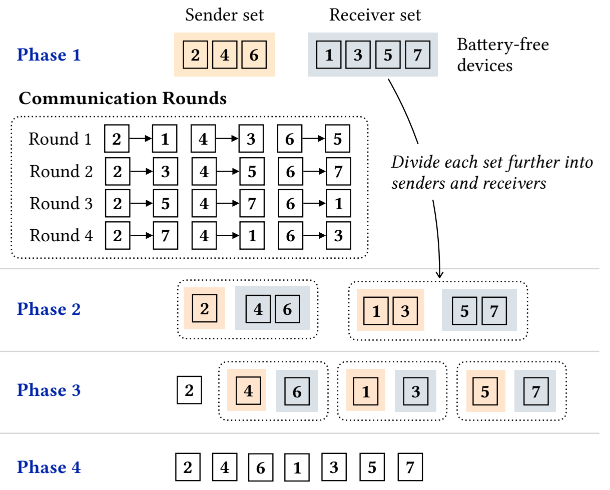


Figure 6: A multi-device communication scheme for Pulsar.

knows the total number of devices and their respective slot allocation, these sets remain consistent across all devices. During the bootstrap phase, devices construct the sender set with even-slot devices and the receiver set with odd-slot devices. Within each phase, communication proceeds through rounds, where each device in the sender set communicates with every device in the receiver set, iteratively. Importantly, within each round, every sender device communicates exclusively with a new receiver device, effectively minimizing collisions. This is achieved by establishing a global ordering of receiver devices to visit, and assigning a unique initial offset to every sender device in this ordered set, which is then adjusted in every round. In the subsequent phase, each set in the current phase is further divided into two sets of senders and receivers, where devices in those new sets communicate. Multiple parallel rounds of communication can occur within one phase between the corresponding sender and receiver sets. The all-to-all communication protocol ends in the phase ($\log_2 N$) when all the sets contain only one device each.

Despite its structured design, this scheme remains susceptible to collisions, particularly because devices making faster progress can outpace other devices and advance through rounds and phases. This susceptibility arises from the underlying protocol for slot jumping, where we let senders jump to the receiver's allocated slots for communication. Hence, the protocol allows multiple senders to be simultaneously active in the receiver's slot, thereby increasing the likelihood of collisions [10], especially when communication progresses asynchronously. To address this issue, we propose an inversion of the protocol for slot jumping, where senders stay in their allocated slots while receivers jump to the sender's slots. This inversion ensures that only one sender is active in any given time slot.

The sender employs broadcasts specifically targeting the intended recipient, while receivers ensure that only the designated recipient responds if active in the same time slot. Here, both senders and receivers need to know their counterparts and the sequence of communication. This is a reasonable assumption in scheduled communication schemes such as ours, where sender-receiver pairs and their communication sequence are predetermined.

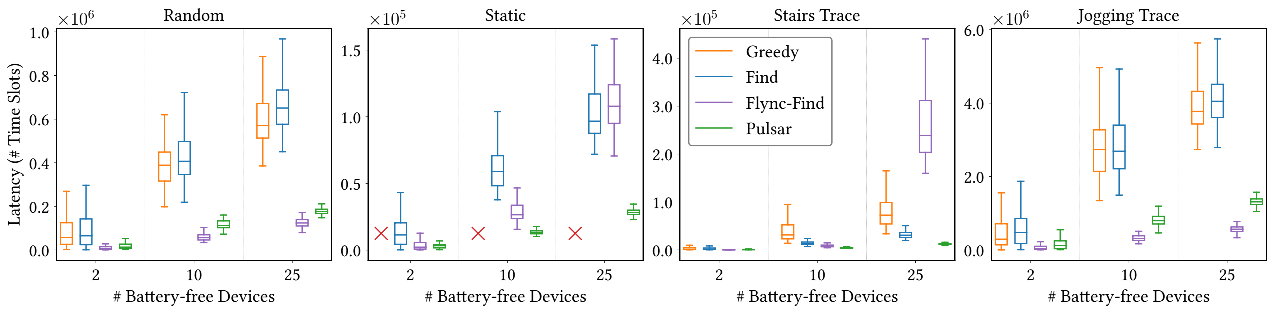


Figure 7: Communication latency of Pulsar compared to Greedy, Find, and hardware-enhanced Flync-Find.

3.5 Collision Handling

While our multi-device communication scheme is collision-free owing to the careful protocol design, Pulsar is not completely collision-free. Collisions may still occur during the initial coordinator discovery stage. The collision rate increases with the number of BF devices in the system. The situation is exacerbated when devices with homogeneous charging times become active in the same slot since these devices will wake up in synchrony, resulting in persistent collisions. To address this persistent collision problem, we propose a technique called *random cycle skipping*. Here, each device randomly decides whether to skip for additional slot distribution cycles by adjusting its sleeping delay as $d_k = d_k + n \times T_{dist}$, with $n = 1$ in this paper. Despite this optimization, collisions persist, albeit not persistently, highlighting the need for further research in collision detection and mitigation techniques.

4 PRELIMINARY EVALUATION

We implement and evaluate Pulsar with simulations and on a prototype built from commodity hardware. We compare Pulsar with the following: Greedy (passive wake-ups with no delay) and Find (active wake-ups with random delay). For completeness, we also include results from Flync-Find with its periodic interval set to 10 slots which enhances Find with special hardware and an external energy signal. The key performance metric is *communication latency* defined as the total time it takes to complete the first round of all-to-all communication, including the discovery and synchronization with the coordinator. To emulate dynamic charging times, we utilize synthetic traces as well as publicly available traces [6].

4.1 Simulation Studies

Setup. We implement our simulator in Python, where BF devices and the coordinator start at random offsets on the global time scale. We conduct experiments on networks comprising 2, 10, and 25 BF devices with one coordinator operating at fixed duty cycles β of 1/4, 1/15, and 1/25, respectively. A slot is set to one millisecond long. We consider four charging conditions: (1) *Random*, which draws charging times uniformly from the range of [100, 500] slots, (2) *Static* and homogeneous, with a charging time of 100 slots, (3) *Stairs Trace*, with minimum, maximum and median charging times of 32, 138, and 43, respectively, and (4) *Jogging Trace*, with minimum, maximum, and median as 84, 68627 and 675. We run simulations for 100 iterations with a maximum simulation time of 10M slots, and we set the probability of random skipping in Pulsar to 0.5.

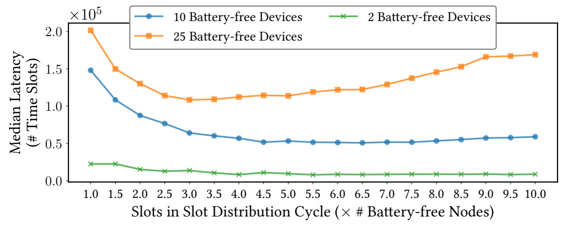


Figure 8: The performance of Pulsar under varying numbers of slots in the slot distribution cycle.

Communication latency. Figure 7 shows the overall performance results. Under Random and Jogging conditions, Find performs similarly to Greedy, confirming our observation that Find is susceptible to heterogeneous and dynamic conditions. Compared with Find, Pulsar achieves significant latency reduction ranging from 2.5× (Stairs, 25 devices) to 5.6× (Jogging, 2 devices), demonstrating its consistently good performance. Flync-Find achieves good performance, even slightly better than Pulsar sometimes, mainly owing to the help of special hardware and an external energy signal. However, Flync-Find may suffer from extremely poor performance in Static and Stairs conditions. Diving into the results we observe that this happens when Find draws a random delay too often within the Flync interval, causing unnecessary collisions constantly when both the charging time and the variability are relatively low.

We would like to point out that the above comparison with Flync-Find is not completely fair to Pulsar since Flync-Find is a discovery protocol and does not have the capability of structured communication as Pulsar does. Currently, our multi-device communication scheme instructs devices to communicate following a specific order and pattern without collisions, while Flync-Find allows devices to discover each other randomly with collisions. This explains why Pulsar slightly underperforms Flyn-Find in some cases. We will explore more efficient communication schemes in future work.

Impact of slot distribution cycle length T_{dist}. Pulsar further minimizes the communication latency for higher values of T_{dist} as shown in Figure 8. Intuitively, increasing T_{dist} boosts the chance of device encounters. Ideally, when T_{dist} is greater than the maximum charging time of all devices, each device should wake up in every slot distribution cycle. However, this ideal condition significantly reduces the duty cycle of BF devices and increases collision rates in large networks, as exhibited for 25 devices in Figure 8. Hence, seeking optimal values of T_{dist} and T_{coor}—independent of each other—to balance the trade-off between the duty cycle of BF devices and the chance of device encounters is an important future work.

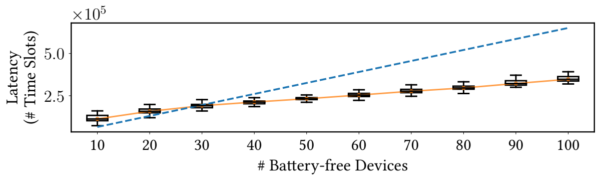


Figure 9: Scalability of Pulsar under random charging conditions. The ideal linear scaling line (shown as the dashed line) is normalized against the two-device case.

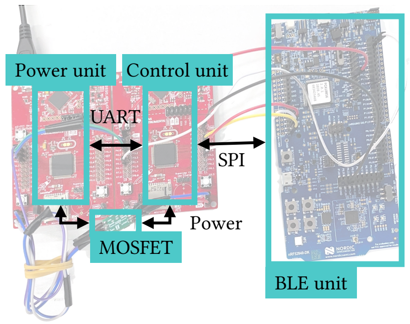


Figure 10: System prototype of a BF device in Pulsar.

Scalability. We evaluate the scalability of Pulsar by varying the number of devices (up to 100) in the network. Figure 9 shows that the median latency grows sub-linearly as device count increases, compared to the ideal scaling line normalized against the two-device performance. Although the collision rate during the initial discovery phase increases with device count, all-to-all communication between devices seems to dominate the time which scales sub-linearly owing to our parallel communication rounds.

4.2 Prototype-based Experiments

We implement a prototype of two BF devices with one coordinator. The BF device is emulated by using two TI-MSP430FR5994 serving as the power supply and communication control units, one NORDIC-NRF52840 as the BLE communication unit, and a MOSFET to control the power supply. The power unit emulates the charging times of BF devices by playing synthetic or real-world charging traces and uses such signals to control the MOSFET to create the charging patterns on the control unit. The control unit then follows the charging cycles and instructs the BLE unit to communicate with other devices for discovery and synchronization. While the noise on the real hardware platform can affect the performance of different approaches especially when it comes to slot alignment, we observe similar performance gains of Pulsar to the simulation results. In particular, under the Stairs condition and across 10 discoveries, Find requires on average 625 slots while Pulsar requires only 173 slots, resulting in a performance gain of 3.6 times.

5 DISCUSSION AND FUTURE WORK

Scalability. The major advantage of BF IoT is significantly reduced maintenance costs and improved sustainability. Introducing coordinator devices should not defeat that. The main question becomes how we can scale the system without introducing too many coordinators since these coordinators are battery-powered and require maintenance. Through our scalability experiments, we show that a single coordinator in Pulsar can already support tens even hundreds of BF devices, at the cost of sublinear latency increase. The exact number of coordinators needed for a real-world deployment depends on the specific requirements concerning maintenance overheads and communication latency. This concerns the setting of system parameters like the duty cycle of the coordinator and the slot distribution cycle. A thorough tradeoff analysis is an interesting direction for future exploration.

Reliability. The coordinator introduces a single point of failure in the system, harming reliability. There are two potential approaches to improving the reliability of Pulsar: (1) We can deploy backups to the coordinator, which maintain heartbeats with the leader coordinator and switch to the leader role when a failure of the current coordinator is detected. The coordinator only maintains soft state so no state synchronization between the coordinator replicas is needed. The BF devices, when losing contact with the coordinator, will start over with the discovery of the (new) coordinator. (2) We could implement a hybrid system where BF solutions like Find serve as a fallback, albeit less efficient, when the coordinator fails and before they are repaired.

More complex communication schemes. Currently, we have only designed and implemented communication schemes for two-device direct communication and multi-device all-to-all communication. While these two schemes can cover a considerable portion of deployment scenarios, other communication schemes (e.g., multicast many-to-many, or more ad-hoc communication patterns) are needed for more complex IoT systems. Pulsar generally supports different communication schemes as long as collisions are managed in these schemes by design.

Modelling communication time. To systematically analyze the benefits and costs of Pulsar compared with other approaches, we could build a model capturing the discovery and synchronization time under a given communication scheme. This model would provide us the opportunity to explore the setup of different parameters and their impact on the overall system performance, thus contributing to deployment decision making.

6 CONCLUSION

We propose Pulsar, a system for facilitating the discovery and synchronization of battery-free devices in IoT systems. Our main motivation is to introduce a certain level of certainty into the system to combat the dynamic, unpredictable charging times of battery-free devices. Pulsar achieves this goal by introducing a battery-powered device among battery-free devices, which serves as a coordinator for battery-free devices to ensure structured communication among them. We present two structured communication schemes for Pulsar and show that through both simulations and prototype-based experiments Pulsar achieves consistently good performance regardless of the charging time variability, without the support of special hardware or environmental conditions as in existing works.

ACKNOWLEDGEMENT

This work has been partially funded by the Dutch Research Council (NWO) grant OCENW.XS22.1.135. Gaosheng Liu is funded by the China Scholarship Council (CSC) fellowship.

REFERENCES

[1] Mikhail Afanasov, Naveed Anwar Bhatti, Dennis Campagna, Giacomo Caslini, Fabio Massimo Centonze, Koustabh Dolui, Andrea Maioli, Erica Barone, Muhammad Hamad Alizai, Junaid Haroon Siddiqui, and Luca Mottola. 2020. Battery-less zero-maintenance embedded sensing at the mithræum of circus maximus. In *ACM Conference on Embedded Networked Sensor Systems (SenSys)*. 368–381.

[2] Jasper de Winkel, Haozhe Tang, and Przemyslaw Pawelczak. 2022. Intermittently-powered bluetooth that works. In *ACM MobiSys*.

[3] Vishal Deep, Mathew L. Wymore, Alexis A. Aurandt, Vishak Narayanan, Shen Fu, Henry Duwe, and Daji Qiao. 2021. Experimental Study of Lifecycle Management Protocols for Batteryless Intermittent Communication. In *IEEE International Conference on Mobile Ad Hoc and Smart Systems (MASS)*. 355–363.

[4] Kai Geissdoerfer and Marco Zimmerling. 2021. Bootstrapping Battery-free Wireless Networks: Efficient Neighbor Discovery and Synchronization in the Face of Intermittency. In *USENIX Symposium on Networked Systems Design and Implementation (NSDI)*. 439–455.

[5] Kai Geissdoerfer and Marco Zimmerling. 2022. Learning to Communicate Effectively Between Battery-free Devices. In *USENIX Symposium on Networked Systems Design and Implementation (NSDI)*. 419–435.

[6] Marco Geissdoerfer, Kai; Zimmerling. 2022. Time-synchronized Energy Harvesting Traces. (2022). https://doi.org/10.5281/zenodo.6383042

[7] Yu Gu, Ting Zhu, and Tian He. 2009. ESC: Energy Synchronized Communication in Sustainable Sensor Networks. In *IEEE International Conference on Network Protocols (ICNP)*. 52–62.

[8] Saptarshi Hazra, Fehmi Ben Abdesslem, and Thiemo Voigt. 2023. Poster Abstract: Battery-free Neighbor Discovery. In *ACM International Conference on Information Processing in Sensor Networks (IPSN)*. 318–319.

[9] Zerina Kapetanovic, Deepak Vasisht, Jongho Won, Ranveer Chandra, and Mark Kimball. 2017. Experiences Deploying an Always-on Farm Network. *GetMobile Mob. Comput. Commun.* (2017), 16–21.

[10] Philipp H. Kindt and Samarjit Chakraborty. 2019. On optimal neighbor discovery. In *ACM Special Interest Group on Data Communication (SIGCOMM)*. ACM, 441–457.

[11] Vito Kortbeek, Kasim Sinan Yildirim, Abu Bakar, Jacob Sorber, Josiah D. Hester, and Przemyslaw Pawelczak. 2020. Time-sensitive Intermittent Computing Meets Legacy Software. In *ACM Architectural Support for Programming Languages and Operating Systems (ASPLOS)*. 85–99.

[12] Gaosheng Liu and Lin Wang. 2021. Self-Sustainable Cyber-Physical Systems with Collaborative Intermittent Computing. In *e-Energy '21: ACM International Conference on Future Energy Systems*. 316–321.

[13] Gaosheng Liu and Lin Wang. 2023. Routing for Intermittently-Powered Sensing Systems. In *IEEE International Performance, Computing, and Communications Conference (IPCCC)*. 274–282.

[14] Gaosheng Liu and Lin Wang. 2024. Data On the Go: Seamless Data Routing for Intermittently-Powered Battery-Free Sensing. *CoRR* abs/2402.00872 (2024).

[15] Gaël Loubet, Alexandru Takacs, Ethan L. W. Gardner, Andrea De Luca, Florin Udrea, and Daniela Dragomirescu. 2019. LoRaWAN Battery-Free Wireless Sensors Network Designed for Structural Health Monitoring in the Construction Domain. *Sensors (Basel, Switzerland)* 19 (2019). https://api.semanticscholar.org/CorpusID:88481723

[16] Kiwan Maeng and Brandon Lucia. 2020. Adaptive low-overhead scheduling for periodic and reactive intermittent execution. In *ACM SIGPLAN International Conference on Programming Language Design and Implementation (PLDI)*. 1005–1021.

[17] Amjad Yousef Majid, Patrick Schilder, and Koen Langendoen. 2020. Continuous Sensing on Intermittent Power. In *ACM/IEEE International Conference on Information Processing in Sensor Networks (IPSN)*. 181–192.

[18] Md. Majharul Islam Rajib and Asis Nasipuri. 2018. Predictive Retransmissions for Intermittently Connected Sensor Networks with Transmission Diversity. *ACM Trans. Embed. Comput. Syst.* (2018), 12:1–12:25.

[19] Sivert T. Sliper, Domenico Balsamo, Nikos Nikoleris, William Wang, Alex S. Weddell, and Geoff V. Merrett. 2019. Efficient State Retention through Paged Memory Management for Reactive Transient Computing. In *ACM Design Automation Conference (DAC)*. 26.

[20] Milijana Surbatovich, Naomi Spargo, Limin Jia, and Brandon Lucia. 2023. A Type System for Safe Intermittent Computing. *Proc. ACM Program. Lang.* 7, PLDI (2023).

[21] Deepak Vasisht, Zerina Kapetanovic, Jongho Won, Xinxin Jin, Ranveer Chandra, Sudipta N. Sinha, Ashish Kapoor, Madhusudhan Sudarshan, and Sean Stratman. 2017. FarmBeats: An IoT Platform for Data-Driven Agriculture. In *USENIX Symposium on Networked Systems Design and Implementation NSDI*. 515–529.