No Battery, No Problem: Challenges and Opportunities in Batteryless Intermittent Networks

Shen Fu, Vishak Narayanan, Mathew L. Wymore, Vishal Deep, Henry Duwe, Daji Qiao
Electrical and Computer Engineering, Iowa State University, Ames, IA, USA
{shenfu, vishakn, mlwymore, vdeep, duwe, daji}@iastate.edu

***Abstract*—The emergence of the Internet of Things (IoT) brings a new paradigm of ubiquitous sensing and computing. Yet as an increasing number of wireless IoT devices are deployed, powering them with batteries becomes expensive and unsustainable. Batteryless systems harvesting energy from the ambient environment offer a promising solution to this problem. However, due to the unpredictability of the ambient energy sources and the relatively weak harvesting strength, these systems may operate intermittently, presenting a series of unique challenges above and beyond the challenges of traditional duty-cycled networks. In this article, we present and discuss the challenges and opportunities in batteryless intermittent networks. We make the case for, and propose, the first formal intermittency-aware network stack. Finally, we present future research directions for batteryless intermittent networks, with the expectation that research of this type can pave the way for batteryless intermittent networks as the next generation of ubiquitous IoT devices.**

I. Introduction

The Internet of Things (IoT) provides a new paradigm of ubiquitous sensing and computing for many applications, such as agriculture, transportation, environment monitoring, home automation, and healthcare systems. Pervasive IoT deployments could result in trillions of devices in coming decades. Powering these devices with batteries is undesirable as batteries are fragile, have a limited lifetime, and create environmental problems, and may need to be regularly replaced in long-term deployments.

Recently, researchers have begun developing batteryless systems in which nodes harvest all of their energy from sources such as radio frequency (RF), thermal, solar, and vibrations, and store it in a comparatively small-capacity capacitor (either ceramic, electrolytic, or super-capacitor). Such batteryless systems increase the feasibility of large-scale, low-maintenance deployments of IoT devices. However, due to the uncontrollable nature of the ambient energy sources and the weak harvesting strength (compared to operating power), these systems face the problem of *intermittency*—they regularly run out of energy and shut down (or *die*). This may be a total shut down at the hardware level (*hard* intermittency), or a deep sleep mode (*soft* intermittency).

Intermittent operation leads to significant new challenges in sensing, computing, and, particularly, communication. Unlike traditional wireless sensor nodes that directly manage their radios' on and off duty-cycles to achieve direct communication, the intermittent nodes must *themselves* be powered on at the same time to achieve direct communication. However, such coincident on-times may only "naturally" occur with a very small probability in intermittent networks because the nodes only operate for a small fraction of the time.

Much work on intermittent communication to this point has focused on limited communication tasks, such as neighbor discovery [1] or backscatter synchronization radios [2]. However, intermittent networks must be capable of general-purpose communication to not only replace traditional battery-powered systems, but fundamentally enable new applications of ubiquitous sensing systems. In order to accomplish this goal under the constraints of intermittency, the system's entire communication and power-management framework must be rethought together.

In this article, we lay out the challenges and research opportunities in batteryless intermittent networks. To provide context and clearly define a batteryless intermittent system, we first present a brief classification of wireless IoT systems. We then discuss the unique features and challenges of batteryless intermittent nodes, providing the motivation for a general-purpose *intermittency-aware* network stack (netstack). We propose such a netstack and define its relationship to other intermittency-aware components that together form a comprehensive intermittency-management framework. We present results demonstrating the role of our netstack in improving intermittent communication performance. Finally, we discuss a selection of research opportunities framed by our intermittency-aware netstack. The goal of this article is to provide a framework within which researchers can better engage and collaborate in the study and exploration of next-generation batteryless intermittent networks, with a vision toward ubiquitous, zero-maintenance wireless IoT systems.

II. Classification of Wireless IoT Systems

To provide context for our discussion, we first classify wireless IoT systems according to the type and capacity of energy storage used and the amount of energy harvested by these systems. Fig. 1 shows a comparison of various *existing* wireless IoT systems in terms of their nodes' designed *single-charge on-time* versus *normalized harvesting rate*. Single-charge on-time is the length of time a node with a full energy charge can sustain operation without energy harvesting before the energy storage depletes to the point where the node must shut down. Normalized harvesting rate is the average energy harvesting rate divided by average energy consumption rate.

As shown in Fig. 1, four clusters of wireless IoT nodes emerge from a design point of view:

- **[Top-left cluster] Limited-lifetime battery-powered nodes:** These nodes are equipped with a battery, and some of them may have the ability to harvest a limited amount of energy. The design goal of this category is to prolong the node's lifetime before the battery depletes.

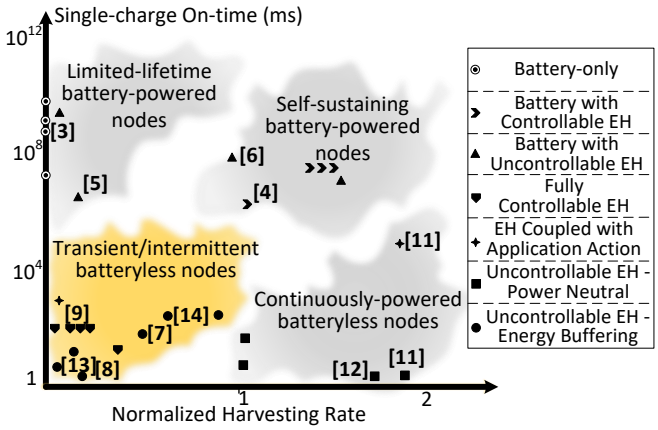

Fig. 1. A taxonomy of wireless IoT nodes. Nodes are classified into clusters based on the type and capacity of energy storage and normalized harvesting rate. Amount and type of control over any energy harvesting are also noted.

- **[Top-right cluster] Self-sustaining battery-powered nodes:** These nodes can generally harvest energy on the same order of magnitude as consumed energy over a long period of time. The main design goal of these systems is to ensure self-sustainability and avoid battery replacement. Due to the relatively large energy capacity of their batteries, these nodes are robust to instability of the harvesting energy source.
- **[Bottom-right cluster] Continuously-powered batteryless nodes:** These nodes are designed to work without a battery but be continuously-powered by an energy harvesting source. The key design goal of these systems is to avoid or minimize total shutdown of the nodes. This could be accomplished via dynamic task scheduling algorithms or application-specific hardware design using techniques such as digitally controllable cores and scalable, dynamically suppressed circuit components.
- **[Bottom-left cluster] Transient/intermittent batteryless nodes:** These nodes have no battery and have little energy storage relative to their active energy consumption, and thus face intermittency. System designers must consider several challenges related to intermittency, including limited communication opportunities and potentially regular loss of volatile state.

Below, we characterize the design considerations of the above clusters in greater detail, including the implications of the amount of available energy and level of control over energy management.

A. Battery-powered Systems

In battery-powered systems, batteries supply nodes with a constant energy source. Nodes are generally deployed with stored energy. Nodes in *Battery-only Systems* are equipped with a non-rechargeable battery (e.g., alkaline, lithium, or dry cell) and, using techniques such as duty-cycling, can sustain themselves for long durations (e.g. [3]). Once the battery depletes, the node dies and is not able to recover by itself.

Alternatively, deployed energy can be augmented with additional energy sources. Nodes in *Battery-powered Systems with Controllable Energy Harvesting* harvest energy from a dedicated energy source (e.g., a wireless mobile charger) to provide energy in an on-demand manner or charge a rechargeable battery (e.g., lithium or sodium ion batteries). These systems have a high degree of controllability in energy harvesting (e.g., [4]) and can be found in the top-right cluster of Fig. 1. Conversely, nodes in *Battery-powered Systems with Uncontrollable Energy Harvesting* usually harvest energy from the ambient environment or uncontrollable sources (e.g., solar, RF, or vibration). These systems can be in the top-left cluster where the harvested energy only prolongs the battery life (e.g., [5], used for general-purpose sensing) or in the top-right cluster where the nodes can become self-sustainable through ultra-low power consumption or energy-constrained optimization (e.g. [6], used for tracking wildlife).

B. Batteryless Systems

Nodes in batteryless systems are solely powered by harvested energy, without the large energy store of a battery. In lieu of a battery, these systems use a capacitor/supercapacitor (multi-layer ceramic [7] or electric double-layer electrolytic [8]) as the energy store. Relatively small capacitors are used to decrease the charging time, equivalent resistance, and leakage. We further classify batteryless systems as follows.

1) Batteryless Systems with Fully Controllable Energy Harvesting or Energy Harvesting Coupled with Application Actions: In this category, the energy harvesting process is controlled to at least some extent. It may be fully controlled, as in most traditional radio-frequency identification (RFID) systems [9]. Or it may be coupled with application actions, such as shoes that harvest kinetic energy when a person is walking or running in them [10]. Systems in this category may exist in either of the bottom two clusters shown in Fig. 1.

2) Batteryless Systems with Uncontrollable Energy Harvesting: The nodes in these systems rely completely on an uncontrollable energy harvesting source. They can be further divided into the following two sub-categories:

a) Power Neutral: A system is called *power neutral* if the energy harvesting rate is higher than the energy consumption rate, i.e., the normalized harvesting rate in Fig. 1 is larger than one. Systems in this category are batteryless but continuously-powered while energy is available to the harvester. Example systems include [11], where the energy source is strong enough to sustain the operation of sensor nodes, and [12], where sensor nodes adjust their power consumption dynamically to be below the harvesting power.

b) Energy Buffering: We call a system *energy buffering* if the energy harvesting rate is generally lower than the energy consumption rate when the system is operating. Nodes in these systems inevitably operate in an intermittent manner. Intermittency can take one of two forms. In hard intermittency, when the energy storage depletes, the system *dies* and loses all volatile state; after recharging, the system restarts and boots back to operation. This method of operation can allow a node to function down to very low harvesting rates and has been implemented in a variety of prototypes [13]–[15]. In soft intermittency [1], [2], the node transitions to deep

sleep when the energy store is close to empty. This allows the node to retain volatile state and basic clock functionality while buffering enough energy to resume operation. Soft intermittency aims for power neutrality in an energy buffering system, which works as long as the typical harvesting power is large enough to sustain the system in deep sleep. If the harvesting is unreliable, a soft intermittent system may sporadically die, effectively becoming a hard intermittent system with extended state and clock retention. We can thus view soft intermittency as a special case of hard intermittency, and consider its intermittency-related challenges to be a subset of those of hard intermittency. We therefore focus our remaining discussion on hard intermittency.

III. Hard Intermittency

From the classification in the previous section, we focus on energy-buffering batteryless systems with uncontrollable energy harvesting—i.e., systems with hard intermittency. These systems are appealing because they could be small, inexpensive, robust, sustainable, and effectively zero-maintenance. This makes them ideal for deployment at scale, even in harsh and inaccessible environments, *if the challenges presented by hard intermittency can be effectively overcome*. In this section, we describe hard intermittency in more detail and summarize the unique features and challenges faced by such nodes. We then focus on intermittent communication, motivating the need for the intermittency-aware netstack presented in the following section.

A. Unique Features and Challenges

As discussed in the previous section, hard intermittency is a condition forced upon a batteryless node when the typical energy harvesting rate is smaller than the consumption rate of the system (either on average or sporadically). As a result, nodes cycle between on and off, as illustrated in Fig. 2.

During an *off-time*, a node charges its capacitor. When the voltage of the capacitor reaches the *on threshold*, the node turns on and operates for a limited time (*on-time* in Fig. 2), until the energy store can no longer support operation. Then the node *dies*, with the CPU, sensors, radios, and active clocks becoming completely unpowered. The node remains off until the energy harvesting recharges the capacitor to the on threshold again.

We refer to this repeated on/off (or alive/dead) pattern of the node itself as *lifecycling*. A node's lifecycling behavior can be characterized by its *lifecycle ratio*—the average on-time relative to the average lifecycle period (i.e., on-time plus off-time). This concept of lifecycling is significantly different from the *duty-cycling* used in many traditional sensor or IoT networks, which refers to the repeated on/off pattern of only the radio, the sensing systems, or the computation activity, controlled by the always-alive (though possibly sleeping) node. The sporadic and uncontrollable nature of ambient energy harvesting and limited control over lifecycling result in a fundamentally different design goal for batteryless intermittent systems—*optimizing application performance under lifecycling constraints*—rather than the design goal of traditional duty-cycled systems—*optimizing energy efficiency under application constraints*.

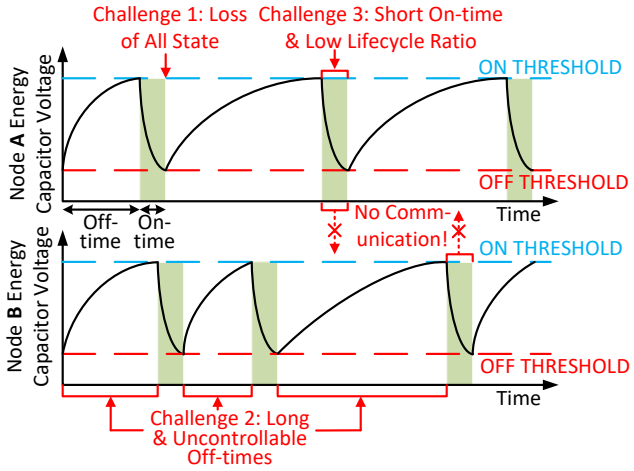

Fig. 2. Lifecycling behavior of batteryless intermittent nodes. The harvested power is much lower than consumption and can change over time, resulting in short on-times and varying, long off-times.

Hard intermittent nodes face several critical challenges:

1) Loss of State during Off-time: When a node dies, it loses all state information not stored in a non-volatile memory, and its clocks cease to oscillate, making it challenging to keep track of tasks or time across off-times. This challenge has received the most attention. To address the loss of task state, example works have used checkpointing or task-based execution to ensure forward progress of applications [16] and have used dynamic energy storage to guarantee that just enough energy is banked to perform the required task. To address the loss of timing information, alternative timing mechanisms called *persistent clocks* [7], [17] have been developed.

2) Long & Uncontrollable Off-times: Pervasive ambient energy sources, such as ambient RF signals, are attractive energy harvesting sources because they require no additional infrastructure support. However, such energy sources are sporadic and uncontrollable. Combined with the generally low energy harvesting rate, the result is long and varied off-times, with little direct control over when on-times occur.

3) Short On-times & Low Lifecycle Ratio: The maximum duration of each on-time is mostly determined by the capacity of the energy storage and the operating power of the node. Since the energy storage capacity is small, and the off-times are long and uncontrollable, the lifecycle ratio of these nodes can be very low (e.g., <1%) and vary over time. This presents many unique challenges in practice, especially for communication, as discussed below.

B. The Case for an Intermittency-aware Netstack

To achieve communication between two traditional (non-intermittent) IoT nodes, their wireless radios need to be on at the same time, which is a classic duty-cycled wireless communication problem. However, as shown in Fig. 2, for intermittent nodes, the situation becomes much more complex—the nodes themselves must be on at the same time for them to

communicate. We refer to such coincident on-time as *overlap*. Applying the challenges of hard intermittency to intermittent communication, we observe that:

- on-times may be several orders of magnitude smaller than off-times.
- a node cannot arbitrarily turn itself on in order to communicate.
- if it runs out of energy, a node cannot prevent itself from dying in order to communicate.
- the loss of volatile state affects constructs such as the packet queue, neighbor table, and acknowledged packets.
- the limited timing information across off-times may not be sufficient for traditional timer-based communication protocols.

In short, intermittent nodes are connected by *intermittent links* that depend on overlap. Natural overlaps are likely to be rare, and the ability of nodes to arrange overlaps is limited. This makes direct intermittent communication inherently difficult, and we observe that the intermittent communication problem can be framed as *maximizing the probability of overlap*.

To the best of our knowledge, there are no tools or frameworks from traditional IoT systems sufficient for addressing this problem. Therefore, new tools are needed—in particular, an *intermittency-aware netstack* that interacts with other intermittency-aware components within the node in order to enact a comprehensive intermittency-management strategy. A well-defined netstack and surrounding components will allow researchers to develop modular, protocol-based solutions to the problems of intermittency, in line with the research process already successfully used in the WSN and IoT communities. In order to design for the worst case and provide a unified framework, such a netstack must provide a means for addressing all of the challenges of hard intermittency. The netstack should then allow alternative intermittency-management solutions (e.g. soft intermittency or always-available wakeup or backscatter radios) to be mapped to a subset of its layers; optimizing such mappings is a future research direction.

Finally, while a node under hard intermittency does not have direct control over its lifecycling, it may have some indirect control, for example, via manipulation of its energy consumption patterns. We have shown in prior work [13], [18] that this limited control can increase the probability of overlap; however, these decisions affect the entire node, not just communication, and need to be made as part of a comprehensive intermittency-management strategy. Therefore, an intermittency-aware netstack should be capable of interfacing with components implementing such a strategy.

IV. An Intermittency-aware Netstack

To address the challenges in intermittent communication laid out in the previous section, an intermittency-aware netstack is needed. As shown in Fig. 3, such a netstack can coexist with other operating system modules and adaptations addressing more general intermittency issues. In this section, we present our proposed intermittency-aware netstack. We then present results demonstrating the effectiveness of our netstack, and discuss future research directions based on these netstack concepts.

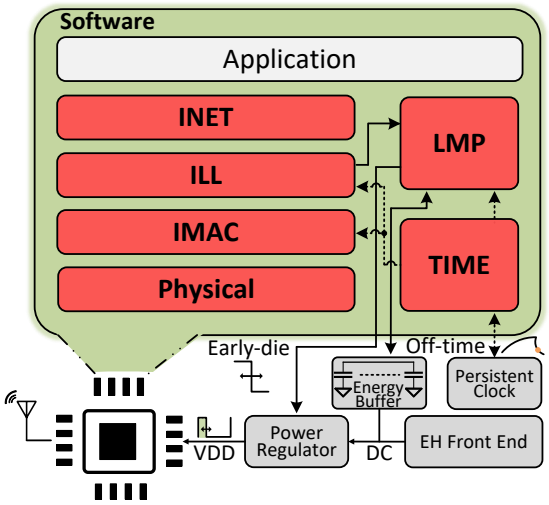

Fig. 3. Our proposed intermittency-aware network stack, along with other intermittency-aware components of the node. Components of the comprehensive intermittency-management framework are shown in red.

A. Overview

Software and hardware for networking have long been conceptually organized into stacks, where each layer has well-defined functions for outgoing (down-the-stack) and incoming (up-the-stack) data. Our proposed intermittency-aware netstack, shown in Fig. 3, resembles the relatively simplified netstacks of duty-cycled WSNs, with intermittency-specific adaptations and a new intermittency-specific layer. Here, we review our proposed layers and describe what unique functions they may perform in an intermittent scenario.

1) Physical Layer: Our netstack is intended to enable general-purpose communication, including with commercial, off-the-shelf active radios, like those used in WSNs and the IoT. That said, the extreme energy demands of intermittent systems are conducive to the use of alternative low-power communication technologies, such as backscatter or wakeup radios. Thus, by default, the other layers of our netstack are agnostic to the choice of physical layer.

2) Intermittent Media Access Control (IMAC) Layer: The IMAC manages access to the wireless channel. Key differences between an IMAC and a traditional wireless MAC are:

- There is not necessarily a practical bound on the length of time a node may need to wait for a receiver to become available.
- This length of time is usually longer than a single on-time for the node.

These differences can significantly affect design goals and approaches. For example, a traditional MAC may regularly delay communication, e.g., for collision resolution. But in an IMAC, delayed communication is often *failed* communication, because the nodes may die at any moment. While many existing low-power MAC protocols may be able to be adapted into IMAC protocols, we believe these shifted design goals will also give rise to novel protocols.

3) Intermittent Link Layer (ILL): Sitting above the IMAC layer, the ILL is a new, intermittency-specific layer that is responsible for managing and adapting to the intermittent nature of links between neighboring nodes. The first function of the ILL is to *maintain communication state across off-times*, similar to how intermittent checkpointing systems maintain computation state across off-times. To achieve this, the ILL must do the following:

- *Persist outgoing packets until they are successfully sent.* To accomplish this, we have the ILL queue outgoing packets in non-volatile memory (and do not necessarily queue packets at any lower layer).
- *Guarantee incoming packets are processed.* To accomplish this, we require an ILL-level acknowledgement for all data packets. When a data packet is received by the lower layers, it is handed to the ILL, which writes it to non-volatile memory *before* generating and sending an ACK. The sender does not remove the packet from its ILL queue until it has received an ILL-level ACK. This order of operations guarantees that either the receiver will be able to process the packet, or that the packet will be retried (if applicable) by the sender.
- *Resume communication activities after an off-time.* Upon rebooting after an off-time, the ILL should determine what, if any, communication activities should occur. A simple action could be to resume sending the packet that was being sent at the end of the previous on-time. More advanced actions could include filtering stale packets (e.g., those not relevant if not sent in the same on-time) from the queue, or targeting a new receiver that is expected to be on now.

The second function of the ILL is to *track intermittency of neighbors*. This includes discovering and maintaining information about the lifecycling of neighbors, such as the historical time between overlaps with a given neighbor. This information can be added to a neighbor table that can be referenced by other layers (e.g., the network layer, for optimizing intermittent routes) or other intermittency-aware components.

The final function of the ILL is to *advocate for spending energy on communication*. When considering a comprehensive intermittency-management approach, the node must divide its limited available energy between various tasks, such as computation, sensing, and communication. The ILL is the primary intermittency-aware component of our netstack, and as such, it is the natural interface between the netstack and other intermittency-aware components of the node, including those performing intermittency management.

As an example, Fig. 3 shows a prototypical node with a Lifecycle Management Protocol (LMP) [13], [18], an intermittency-aware component that exerts (limited) control over the lifecycling behavior of the node in pursuit of certain design goals. For instance, the ILL can request that the LMP "early-die," purposefully shortening the on-time and banking energy to shorten the following off-time. This increases the frequency of on-times, potentially decreasing the expected time between overlapping on-times of neighboring nodes (shown analytically and experimentally in [13]). Then, when communication does occur, the ILL may want to maximize the communication opportunity and request that the LMP extend the on-time. A more advanced ILL might dynamically request different configuration parameters for the LMP based on the current communication context. This separation of concerns preserves the focus of the netstack, since the LMP typically interfaces with non-networking hardware. It also allows the LMP to act as a more general intermittency-management component.

4) Intermittent Network Layer (INET): The INET is responsible for constructing and managing *intermittent paths*, i.e., paths built using intermittent links. Such paths are reminiscent of the lossy or energy-aware paths of prior IoT and WSN work, but pushed to the extreme, to the extent that the design goals of an INET may differ from traditional network layers. For example, instead of maximizing throughput or minimizing end-to-end delay, an INET may wish to maximize delivery probability within a delay bound. Therefore, we expect that novel INET protocols could provide significant gains in intermittent networking.

B. Evaluation of Netstack

To demonstrate the effectiveness of our proposed intermittency-aware netstack, we present results from an experimentally-validated SimPy-based event-driven simulator that implements our netstack, along with the LMP component, as a modular framework. The simulator allows us to evaluate the individual contributions to intermittent communication performance of some example protocols at each layer. We evaluate the following combinations of protocols:

- **Naive** represents a traditional duty-cycled WSN netstack with no adaptations for intermittency. Communication state, including the packet queue, is not saved across off-times.
- **Naive-LMP2** adds a simple LMP mechanism to Naive. The LMP shortens each on-time to the length of two round-trip communications (i.e. two *slots*, or 5 ms in our simulator configuration).
- **Minimal** is a minimal ILL that maintains communication state across off-times. Specifically, the packet queue is saved in non-volatile memory, and at the beginning of each on-time, the ILL resumes attempting to send packets from the queue.
- **Minimal-LMP2** adds the two-slot on-time LMP mechanism to Minimal.
- **Full** extends Minimal-LMP2. When communication occurs, the ILL requests that the LMP extend the on-time. When communication finishes, the ILL notifies the LMP to resume normal operation.

Our IMAC is a baseline sender-initiated MAC in which the receiver idly listens and the sender repeatedly transmits the data packet until it is ACKed. The IMAC transmits the same packet until it is successful, or until the node dies. We simulate two nodes, both of which are intermittent (with expected lifecycle ratios of 11% and 7%), with individual off-times varying in a Gaussian manner with a standard deviation

of 30% of the mean. Both nodes generate data traffic via a Poisson point process with a given rate parameter.

We present throughput results for these protocols under a variety of traffic generation rates in Fig. 4. The traffic upper bound shows the expected throughput if all packets are delivered successfully. The lifecycle ratio (LCR) upper bound shows the expected throughput if the more energy-constrained node spends every on-time successfully communicating packets (i.e., a perfect ILL). The results make general sense—median throughput converges to the traffic bound for the best protocols at lower traffic rates. As traffic rates increase, the performance becomes bounded by the LCR, with the gap between median throughput and the LCR bound depending on the effectiveness of the protocol, essentially measuring how much of the available energy is successfully being used for communication.

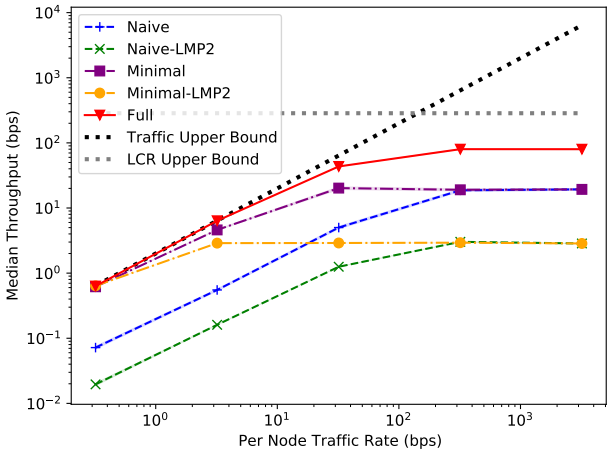

Fig. 4. Total throughput vs. expected per-node traffic rate, for a pair of intermittent nodes. Each data point consists of 40 trials of 10 hours of simulated time. 95% confidence intervals, plotted as shaded areas, are mostly too small to be visible. Note the log scale on both axes.

Beginning with Naive, we can see the individual contributions of the components. Adding only the LMP mechanism to Naive (Naive-LMP2) *decreases* median throughput across the range of traffic rates. While this LMP mechanism increases the frequency of successful communication, without the ILL advocating for extended on-times from the LMP, nodes are only able to send one or two packets per overlapping on-time. Adding a persistent queue to Naive (Minimal) increases throughput at lower traffic rates, but has no effect at higher traffic rates, when the node always has packets to send each on-time even without saving them from prior on-times. Adding the LMP mechanism along with the persistent queue (Minimal-LMP2) improves on Naive's performance at lower traffic rates. But, this protocol quickly becomes LCR-bounded, for the same reasons as Naive-LMP2.

Finally, with the full intermittency-aware netstack working in conjunction with an LMP, Full has the best throughput performance across the range of traffic rates, achieving up to 11.5× the throughput of Naive (at the 3.2 bps traffic rate).

These results demonstrate the need for our proposed intermittency-aware netstack, particularly the ILL and its interactions with other intermittency-aware components in the system. These results also hint at areas ripe for potential research—while our example Full protocol greatly improves on the Naive baseline, there is still a large throughput gap between it and the LCR bound. This means that there is energy in the system for more communication performance.

The intermittency-aware netstack also has a dramatic effect on communication reliability. In Fig. 5, we show a CDF of packet delivery delays at the lowest traffic rate tested. As expected, the Full protocol performs the best, achieving the smallest delays with the least variation. Moreover, packets that are not delivered are shown with infinite delay in the plot. This clearly shows the need for the ILL's first function; without a persistent queue, Naive and Naive-LMP2 fail to deliver well over 80% of the generated traffic. The traffic that they do deliver has an (unfairly) low delay, because this traffic was added to the queue during the same on-time in which it is sent.

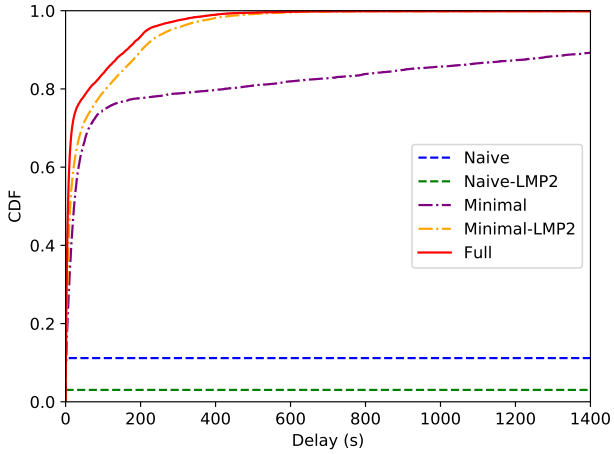

Fig. 5. CDF of packet delivery delay (queue time to acknowledgement time) at an expected traffic generation rate of 0.32 bps. Packets that are not delivered due to queue-full conditions, or (in the case of Naive and Naive-LMP2) loss of queue state, have an infinite delivery delay.

C. Research Opportunities

With the framework of our intermittency-aware netstack, we can identify distinct areas for research in intermittent communication. We believe significant improvements can be made at the IMAC, ILL, and INET layers.

1) IMAC: The IMAC layer needs research in protocol design, from adapting existing low-power MAC protocols to the intermittent case, to designing new protocols using intermittency-specific design goals. Looking beyond unicast communication, the problem of efficient MAC-layer broadcasting to intermittent nodes appears particularly challenging. Coordination between the IMAC and the ILL also poses research questions. For example, outside of normal communication, what lifecycling information of other nodes could the IMAC provide the ILL?

2) ILL: Advanced ILL protocols is an area ripe for research. As shown in the previous section, there is significant room for improvement in the ILL's performance even in the two-node case. When extended to two or more neighbors, the

considerations of the ILL become much more complex. We also envision the ILL making more use of its interface with other intermittency-aware components such as the LMP. For example, the ILL could dynamically reconfigure the LMP to adapt to various communication scenarios. A co-design of ILL and LMP could also be fruitful, and an effective means for integrating novel intermittency management hardware [15] as well.

3) INET: To the best of our knowledge, the effects of hard intermittency on the network layer, and the corresponding solutions, have not been explored at all. We expect that protocols designed specifically for intermittent paths could provide significant benefits, particularly in terms of reliability. Network layer research can also address a variety of different traffic scenarios, including many-to-one data collection, one-to-one routing, one-to-many multicast, and network-layer broadcast or flooding.

4) Cross-layer: Finally, while we believe that codifying an intermittency-aware netstack is an important step in solving the communication problem, we also believe that cross-layer approaches are well-suited for hard intermittency. For example, a network layer building intermittent paths may be interested in the characteristics of the intermittent links composing those paths. An opportunistic routing protocol, of the sort that dynamically selects from available forwarders at the MAC layer, could use a routing metric that takes lifecycling information into account. Such a protocol would effectively cross three layers—INET, ILL, and IMAC.

V. Conclusion

In conclusion, the unique constraints of batteryless intermittent nodes make networking these nodes a new and distinct problem that requires new approaches and frameworks. To this end, in this paper, we have described a intermittency-aware netstack designed to coexist with other intermittency-aware components of a comprehensive intermittency-management framework. Using a simulator calibrated with experimental data from fabricated prototypes, we have shown how the novel components of this intermittency-aware netstack can interact with the intermittency-management framework to greatly improve communication performance under the constraints of hard intermittency. Finally, we have enumerated some future research directions and opportunities for the proposed netstack.

References

[1] K. Geissdoerfer *et al.*, "Bootstrapping battery-free wireless networks: Efficient neighbor discovery and synchronization in the face of intermittency," in *18th USENIX Symposium on Networked Systems Design and Implementation (NSDI 21)*, 2021, pp. 439–455.

[2] A. Torrisi *et al.*, "Reliable Transiently-Powered Communication," *IEEE Sensors Journal*, vol. 22, no. 9, pp. 9124–9134, 2022.

[3] M. Konijnenburg *et al.*, "28.4 a battery-powered efficient multi-sensor acquisition system with simultaneous ecg, bio-z, gsr, and ppg," in *IEEE ISSCC*, 2016.

[4] S. Nabavi *et al.*, "Efficient and Easy to Fabricate Silicon-Based Planar Micro-Coils for Wireless Power Transfer Applications," *IEEE Sensors Journal*, vol. 22, no. 3, pp. 1980–1989, 2021.

[5] Y. Li *et al.*, "Developing a miniature energy-harvesting-powered edge device with multi-exit neural network," in *2021 IEEE International Symposium on Circuits and Systems (ISCAS)*. IEEE, 2021, pp. 1–5.

[6] P. Sommer *et al.*, "Energy-and mobility-aware scheduling for perpetual trajectory tracking," *IEEE Transactions on Mobile Computing*, vol. 19, no. 3, pp. 566–580, 2019.

[7] V. Deep *et al.*, "HARC: A heterogeneous array of redundant persistent clocks for batteryless, intermittently-powered systems," in *2020 IEEE Real-Time Systems Symposium (RTSS)*. IEEE, 2020, pp. 270–282.

[8] A. Colin *et al.*, "A reconfigurable energy storage architecture for energy-harvesting devices," in *ACM ASPLOS*, 2018.

[9] M. Karimi *et al.*, "Real-Time Task Scheduling on Intermittently Powered Batteryless Devices," *IEEE Internet of Things Journal*, vol. 8, no. 17, pp. 13 328–13 342, 2021.

[10] Q. Huang *et al.*, "Toward battery-free wearable devices: The synergy between two feet," *ACM Transactions on Cyber-Physical Systems*, vol. 2, no. 3, pp. 1–18, 2018.

[11] S. Bose *et al.*, "A batteryless motion-adaptive heartbeat detection system-on-chip powered by human body heat," *IEEE Journal of Solid-State Circuits*, vol. 55, no. 11, pp. 2902–2913, 2020.

[12] A. Roy *et al.*, "A 6.45 μw self-powered soc with integrated energy-harvesting power management and ulp asymmetric radios for portable biomedical systems," *IEEE Transactions on Biomedical Circuits and Systems*, vol. 9, no. 6, pp. 862–874, 2015.

[13] V. Deep *et al.*, "Experimental study of lifecycle management protocols for batteryless intermittent communication," in *2021 The 18th IEEE International Conference on Mobile Ad-Hoc and Smart Systems*, 2021.

[14] V. Narayanan *et al.*, "BOBBER: A prototyping platform for batteryless intermittent accelerators," in *Proceedings of the 2023 ACM/SIGDA International Symposium on Field Programmable Gate Arrays*, 2023, pp. 221–228.

[15] M. Gshash *et al.*, "RF energy harvester with constant off-time charger for batteryless devices," in *2023 IEEE 66th International Midwest Symposium on Circuits and Systems (MWSCAS)*, 2023.

[16] A. Y. Majid *et al.*, "Dynamic task-based intermittent execution for energy-harvesting devices," *ACM Trans. Sen. Netw.*, vol. 16, no. 1, Feb. 2020. [Online]. Available: https://doi.org/10.1145/3360285

[17] J. de Winkel *et al.*, "Reliable timekeeping for intermittent computing," in *ACM ASPLOS*, 2020.

[18] M. L. Wymore *et al.*, "Lifecycle management protocols for batteryless, intermittent sensor nodes," in *IEEE IPCCC*, 2020.

Biographies

Shen Fu received his Ph.D. degree in Computer Engineering from Iowa State University (ISU). His research interests include cyber security, applied machine learning, and IoT.

Vishak Narayanan received his B.Tech. degree in Electronics and Communication Engineering from SCMS School of Engineering and Technology. He is currently pursuing his Ph.D. degree in Computer Engineering at ISU, with research interests including computer systems architecture, embedded systems, and VLSI.

Mathew L. Wymore received his Ph.D. degree in Wind Energy Science, Engineering and Policy, co-major in Computer Engineering, from ISU. He is an Assistant Teaching Professor in the Department of Electrical and Computer Engineering (ECpE) at ISU. His research interests include wireless sensor networks and intermittent systems.

Vishal Deep Vishal Deep is currently a Ph.D. student in Computer Engineering at ISU. He received his M.S. degree from California State University, Fresno. His research interests include embedded systems, IoT, and computer architecture.

Henry Duwe received his Ph.D. in Computer Engineering from the University of Illinois at Urbana-Champaign. He is an Assistant Professor in the ECpE Department at ISU. His research interests include computer architecture, compilers, and design automation.

Daji Qiao received his Ph.D. degree in Electrical Engineering: Systems from the University of Michigan, Ann Arbor. He is a Professor in the ECpE Department at ISU. His research interests include wireless networking and mobile computing, sensor networks and IoT, and cyber security.