# Energy-Efficient System Design for IoT Devices*

Hrishikesh Jayakumar, Arnab Raha, Younghyun Kim,
Soubhagya Sutar, Woo Suk Lee, and Vijay Raghunathan
{hjayakum, araha, yhkim1, ssutar, lee992, vr}@purdue.edu

**Abstract— It is projected that, within the coming decade, there will be more than 50 billion smart objects connected to the Internet of Things (IoT). These smart objects, which connect the physical world with the world of computing infrastructure, are expected to pervade all aspects of our daily lives and revolutionize a number of application domains such as healthcare, energy conservation, transportation, *etc*. In this paper, we present an overview of the challenges involved in designing energy-efficient IoT edge devices and describe recent research that has proposed promising solutions to address these challenges. First, we outline the challenges involved in efficiently supplying power to an IoT device. Next, we discuss the role of emerging memory technologies in making IoT devices energy-efficient. Finally, we discuss the potential impact that approximate computing can have in increasing the energy-efficiency of wearables and other compute-intensive IoT devices.**

## I. Introduction

Various industry forecasts project that, by 2020, there will be around 50 billion smart devices connected to the Internet of Things (IoT) helping to engineer new solutions to societal-scale problems such as telemetry, healthcare, home automation, energy conservation, security, wearable computing, asset tracking, maintenance of public infrastructure, *etc*. A majority of these smart devices will lie at the edge of the IoT, bridging the physical world with the world of computing. We refer to these devices as IoT edge devices. A major challenge in realizing the vision of the IoT is the problem of powering these billions of edge devices. Due to the expense, inconvenience, or sheer infeasibility of wiring them, most of these edge devices are expected to be untethered and powered using batteries and/or energy harvesting. Further, stringent constraints on the device form-factor (and hence, the on-board energy storage capacity) exacerbate the problem as most of these IoT edge devices are required to have long operational lifetimes, from a few days to several years.

Fig. 1 shows a taxonomy (defined in [1]) of IoT edge devices according to the power available to them and the longevity requirements. The first class of devices represents wearable devices (*e.g.*, smart watches, fitness monitors, *etc*.), which have a longevity requirement spanning from a few days to weeks. The next group comprises the dozens of set-and-forget (SAF) devices that each person is likely to own (*e.g.*, home automation gadgets, water leak sensors, *etc*.), which are expected to last a couple of years without any user intervention. The next group of devices represents infrastructure and geophysical monitoring systems or IGMS (*e.g.*, wireless sensors that monitor public infrastructure such as bridges, highways, and parking structures), which are semi-permanent devices whose longevity requirement exceeds several years. The last category consists of transiently powered computers or TPCs (*e.g.*, RFID tags, smartcards, *etc*.), which are batteryless devices that depend solely on remote or ambient power supplies. As Fig. 1 shows, there is considerable heterogeneity in the power availability and longevity requirements of different IoT device types, and as a result, a one-size-fits-all approach to energy optimization will not work for these IoT devices. This paper presents an overview of the challenges involved in the energy-efficient system design of different classes of IoT edge devices and highlights recent research that has proposed promising solutions to address these challenges.

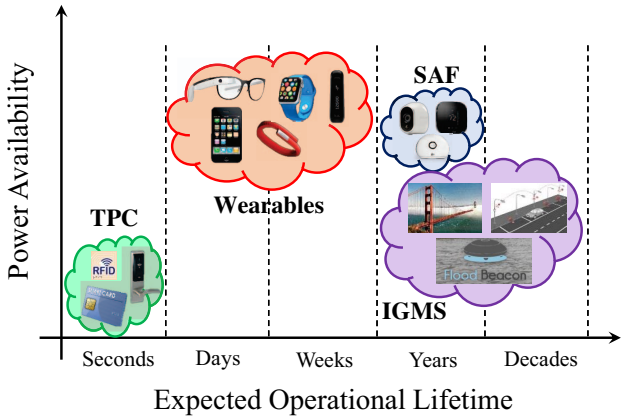


Fig. 1. IoT edge device classification [1]

The rest of the paper is arranged as follows. Section II describes the challenges in supplying energy to IoT edge devices, Section III outlines the challenges and some techniques in improving energy-efficiency of SAF, IGMS, and TPC edge devices, and Section IV describes how approximate computing can improve the energy-efficiency of wearable devices.

## II. Challenges in Efficient Energy Supply

The sheer number of IoT edge devices expected to be deployed in the near future brings with it dauting challenges in reducing the post-deployment maintenance cost. In addition to the cost of new batteries, the maintenance cost of an IoT edge device includes the cost of disposing depleted batteries, labor, and the cost incurred due to system downtime. Addressing this challenge is paramount in order to enable the continued adoption of IoT devices at a brisk pace. The problem is exacerbated in SAF and IGMS devices due to their scale and deployment scenarios. To address this challenge, energy harvesting has emerged as an attractive and increasingly feasible solution that can enable significantly prolonged operational lifetime and in some cases, even self-sustained, near-perpetual operation.

*This work was supported in part by the National Science Foundation (NSF) through grants CNS-0953468 and CCF-1018358.

978-1-4673-9569-4/16/$31.00 ©2016 IEEE

Energy harvesters convert power from ambient sources such as electromagnetic radiation (*e.g.*, sunlight, WiFi signals), thermal gradients, mechanical motion (*e.g.*, vibration), *etc*., into electrical power. The choice of harvesting modality for a particular IoT device is dependent on its operating environment, form-factor constraints, as well as its power budget. Harvesting energy using photovoltaics is the most mature technique among all the different energy harvesting techniques, in part because of its relatively high power-density. IoT devices that have substantial exposure to light are most suited for harvesting energy in this manner (*e.g.*, outdoor environment monitoring [2, 3]). For indoor IoT devices where strong light may not always be available, energy harvesting of RF signals is gaining popularity [4]. RF energy harvesting circuits convert either dedicated RF waves generated for wireless charging [5] or ambient RF waves (*e.g.*, WiFi, television broadcast signals, *etc*.) into electrical energy [6, 7]. Other modalities of energy harvesting include transduction of kinetic energy and thermoelectric energy into electrical energy. Kinetic energy harvesting devices convert mechanical energy produced by the motion or vibration of the human body and machines into electrical energy [8, 9]. On the other hand, thermoelectric generators can be used [10] in locations where a large thermal gradient is available (*e.g.*, hot water pipes). Another technique that has been explored recently is to power certain sub-systems of an IoT edge device (such as a real time clock, microcontroller in idle mode, *etc*.) by scavenging the energy output from sensors when they are idle [11], which otherwise goes wasted.

In addition to the energy harvester, an energy storage unit is an indispensable component in IoT edge devices. Many ambient energy sources have highly dynamic, uncontrollable output (*e.g.*, the energy output from a photovoltaic cell is subject to dynamic variations in solar irradiance levels), making energy storage a key component in the power supply architecture of the device. The properties of the energy storage unit are dictated by both the IoT application and the energy source of the particular IoT device. For example, most wearable devices are periodically recharged by the user. Additionally, these devices are typically of small form-factor. Therefore, high energy density is the most critical requirement for ensuring long lifetime and as a result, lithium-ion batteries are currently the dominant form of energy storage in these devices. On the other hand, in ambiently-powered IoT devices, where even a minuscule amount of energy cannot be squandered, high cycle efficiency or roundtrip efficiency is the critical trait. Supercapacitors are promising energy storage components for such devices that undergo frequent charge-discharge cycles [8, 12].

Despite these efforts to improve the efficiency and reliability of the power supply in IoT devices, it is not always practical or feasible to provide continuous power to the device under limited ambient energy availability and stringent form-factor constraints. Therefore, additional system-level techniques need to be developed for these devices to be resilient and reliable in the face of power loss, as discussed in the following section.

## III. Leveraging Memory for Energy-Efficiency in IoT Edge Devices

As mentioned in Section I, SAF and IGMS devices are expected to have battery lifetimes of several years and, therefore, need to operate at extreme levels of energy efficiency. TPCs have a similar requirement of extreme energy-efficiency due to the fact that they are batteryless. This challenge is further aggravated due to constraints on form-factor (which limit the capacity of on-board energy storage) and deployment location (which limit the type and amount of energy available for harvesting). A key observation in addressing this challenge for SAF and IGMS devices is that they typically operate in heavily duty-cycled mode. This results in the idle mode being responsible for the lion's share of energy consumption in these systems. Most modern microcontrollers (MCUs) address the issue of idle power consumption by providing two kinds of sleep modes, namely, *shallow sleep* (which results in fast wake-up and state retention) and *deep sleep* (which takes longer time to wake-up and does not retain state) that have lower power consumption than the active mode. An MCU enters and exits *shallow sleep* mode with little overhead but keeps on-chip memory elements powered to retain state and, hence, consumes significant power. On the other hand, *deep sleep* consumes nearly zero power but requires the MCU to checkpoint the current system state to the on-chip non-volatile memory (typically flash). Hence, the overall energy cost of entering and exiting *deep sleep* is significant (as flash writes/erases are energy intensive operations), which deters MCUs from entering *deep sleep* for short idle periods. An ideal sleep mode should be as low power as *deep sleep* and yet retain state with the ease of transitioning in and out of *shallow sleep*. Hypnos [13] is a step in this direction and architects a new sleep mode for MCUs that combines the state retention of *shallow sleep* modes with the extremely low power of *deep sleep* modes. It achieves this by scaling the MCU's supply voltage to just above the SRAM's data retention voltage [14, 15] while MCU is in sleep mode.

Another problem faced by SAF, IGMS, and TPC devices that are powered by ambient energy sources is the unreliability in power supply. The uncertainty in ambient energy means that the device can suffer a sudden loss of power. Therefore, executing long-running applications is challenging in these devices due to frequent system reboots. Recent research has addressed the issue of enabling long-running computations in such transiently powered computers (TPCs) [16, 17] by checkpointing system state before an imminent power loss (similar to entering *deep sleep*). Mementos [18] is one such solution that checkpoints system state to the on-chip flash memory. However, the energy intensive erase/write operations of flash eat into the amount of energy available for performing meaningful execution in each power cycle. Advances in memory technologies have seen the emergence of non-volatile memories (NVMs) such as ferroelectric RAM (FeRAM), magnetoresistive RAM (MRAM), resistive RAM (ReRAM), *etc*., that combine the flexibility and endurance of SRAM with the non-volatility of flash, all at a very low power consumption. Quickrecall [19, 20] and Hibernus [21, 22] use microcontrollers integrated with FeRAM, similar to [23], to enable efficient checkpointing in TPCs. Quickrecall utilizes FeRAM as a unified memory, thus enabling *in-situ* checkpointing. Quickrecall also checkpoints *on demand* and, unlike Mementos, does not need to do any checkpointing-related activity periodically. Hibernus uses FeRAM as the on-chip non-volatile memory instead of flash. Although FeRAM is superior to flash, it is slower than SRAM due to inherent device limitations. However, SRAM is volatile and, hence, data present in SRAM needs to be checkpointed on imminent power loss. Therefore, a trade-off exists between checkpointing and the energy consumed for program execution. Ref. [24] aims to dynamically map a program's section to FeRAM and SRAM such that overall energy consumption is minimized.

The research works described above adopt a joint hardware-software approach to checkpointing wherein software (along with hardware) performs the checkpoint operation. Recent interest in state retention for energy harvesting applications has resulted in the emergence of non-volatile processors (NVPs) [25–29], which are hardware-only solutions to state retention. NVPs integrate volatile memory elements with emerging NVMs that enable them to take a snapshot of the system state on power loss and restore it on subsequent power availability. The authors in [30] utilize an NVP to architect a TPC that contains no energy storage or converter. However, by design, NVPs bypass normal boot procedures on power restoration and continue executing instructions where they left off previously. For energy harvesting IoT devices that have little notion of time, continuing to perform a task (such as communication, data sampling, *etc*.) on power restoration is not always functionally correct, as the checkpointed state might be stale, and hence, invalid upon wake-up. Related research also explores different architectural choices for NVPs [31]. However, the impact that different architectures have on *full-system* energy consumption remains to be seen in these devices. Finally, recent work proposes programming models to address checkpointing related consistency issues [32,33] in energy harvesting TPCs utilizing NVPs.

## IV. Approximate Computing: The Next Frontier in Energy-Efficient Computation

Energy-efficient operation is critical for battery-powered wearable devices, especially since they are increasingly being used to execute computationally intensive applications. Many of these applications belong to the domains of recognition, data mining, vision, multimedia, and digital processing, which are inherently error resilient in nature (*i.e.,* the quality of the application's output is resilient to a small amount of noise introduced in the computations or the input data itself). Approximate computing is an emerging design paradigm that can leverage this intrinsic resilience of applications to execute computations approximately, and more efficiently, resulting in unprecedented levels of energy improvement.

Approximate computing has been explored at various layers of design abstraction, spanning circuits, architectures, and software. A large body of the existing work pertains to approximate circuit design [34–40], where approximations have been introduced in specific arithmetic units such as adders [34, 35] and multipliers [39]. Recent work has broadened the scope by proposing techniques for approximating any logic circuit. These methods include simplifying the circuit by injecting stuck-at-faults at certain nodes in the circuit [37], leveraging path activation properties to delete gates in less active portions [40], and exploiting the use of don't cares to simplify circuits using traditional boolean optimizations [38]. Researchers have also proposed architectural techniques for approximate computing [41–43]. For example, Ref. [41] proposes a scalable-effort processor specific to recognition, mining, and synthesis applications, whereas Ref. [42] presents a quality-programmable vector processor applicable to all approximate applications. Ref. [43] proposes micro-architecture extensions to efficiently map critical and approximate portions of applications onto hardware. Finally, approximation techniques at the software-level include high-level transformations such as loop perforation [44], computation skipping [45], and replacing resilient portions of code with a corresponding simpler neural network that approximates the code's function [46].

A key challenge in approximate computing is that the extent to which computations can be approximated varies significantly from application to application, and also across inputs for a single application. This makes quality-configurability (the ability to modulate the energy *vs.* quality trade-off of applications at runtime) and input-adaptive approximation (the ability to modulate the degree of approximation based on the nature of individual inputs) essential for obtaining significant energy savings while retaining output quality at acceptable levels. Towards this end, Ref. [47] demonstrates that statically fixing the degree of approximation in the functional units of an MPEG encoder results in widely varying output video quality. As a solution, the authors designed quality-configurable arithmetic functional units and constructed a runtime framework that can automatically tune the level of approximation in these arithmetic blocks on the basis of internal variables. This results in considerable power savings while also ensuring adherence to the specified quality bound. A more generic approach is presented in Ref. [48], which explores input-adaptive approximations in the context of a key computational pattern, *Reduce-and-Rank* (RnR). RnR forms the core of a wide range of error-resilient applications such as k-Nearest Neighbors, K-Means clustering, MPEG encoder, Generalized Learning Vector Quantization, Image Segmentation, and the Sobel operator. The authors propose two complementary strategies, interleaved reduction-and-ranking and input similarity based approximations for approximating the RnR kernel. These strategies leverage partial reduction outputs and spatial/temporal similarity between inputs, respectively, to identify future computations that are likely to have a low (or no) impact on the output and, therefore, can be approximated.

The error resilient nature of such applications also gives rise to the prospect of designing approximate memories, where the strict constraints on data integrity can be relaxed in exchange for large savings in energy consumption. Approximate memories have been investigated for different memory technologies such as multi-level phase-change memory cells [49], DRAM [50, 51], and SRAM [52]. As an example, Ref. [51] proposes the notion of a quality-aware approximate DRAM where memory pages can be split into a number of quality bins based on the number, location, and nature of bit errors manifesting in each page at reduced refresh rates. Error-resilient data is then systematically allocated to these erroneous pages introducing a controlled level of approximation, which ensures that output application quality remains within acceptable limits.

Finally, we believe that the true potential of approximate computing will be realized only when approximations are performed at a *full-system level* instead of dealing with different subsystems/components in isolation. Controlled levels of approximation can be introduced in the sensing, communication, computation, and memory subsystems of an IoT device, leading to the construction of an *end-to-end approximate system*. Then, the main challenge will then be to synergistically tune the various knobs controlling the accuracy-energy trade-off for each component/sub-system to ensure operation at the most desirable point on the *global* energy *vs.* accuracy curve for the entire IoT device.

## V. Conclusions

This paper presented an overview of the challenges involved

in the energy-efficient system design of different classes of IoT edge devices and highlighted recent research that has proposed promising solutions to address these challenges. In particular, the paper discussed the problems associated with efficiently delivering energy to IoT edge devices, outlined the promising role that emerging memory technologies can potentially play in making these devices energy-efficient, and, finally, described the impact that approximate computing can have on increasing the battery of wearable and similar IoT devices.

## REFERENCES


[1] H. Jayakumar et al. Powering the internet of things. In *ISLPED*, pages 375–380, 2014.

[2] C. Alippi et al. A robust, adaptive, solar-powered wsn framework for aquatic environmental monitoring. *Sensors Journal, IEEE*, 11(1):45–55, 2011.

[3] Bradford Campbell and Prabal Dutta. An energy-harvesting sensor architecture and toolkit for building monitoring and event detection. In *BuildSys*, pages 100–109, 2014.

[4] Sangkil Kim et al. Ambient rf energy-harvesting technologies for self-sustainable standalone wireless sensor platforms. *Proceedings of the IEEE*, 102(11), 2014.

[5] H. Jabbar et al. RF energy harvesting system and circuits for charging of mobile devices. *IEEE T CONSUM ELECTR*, pages 247–253, 2010.

[6] B. Allen et al. Harvesting energy from ambient radio signals: A load of hot air? In *LAPC*, pages 1–4, 2012.

[7] E. Abd Kadir et al. Indoor wifi energy harvester with multiple antenna for low-power wireless applications. In *ISIE*, pages 526–530, 2014.

[8] M. Gorlatova et al. Movers and shakers: Kinetic energy harvesting for the internet of things. In *SIGMETRICS*, pages 407–419, 2014.

[9] Jing-Quan Liu et al. A MEMS-based piezoelectric power generator array for vibration energy harvesting. *Microelectronics Journal*, 39(5):802 – 806, 2008.

[10] Y.K. Ramadass and A.P. Chandrakasan. A battery-less thermoelectric energy harvesting interface circuit with 35 mV startup voltage. *IEEE J SOLID-ST CIRC*, pages 333–341, 2011.

[11] W. S. Lee et al. When they are not listening: Harvesting power from idle sensors in embedded systems. In *IGCC*, pages 1–10, 2014.

[12] R. Vyas et al. A battery-less, energy harvesting device for long range scavenging of wireless power from terrestrial TV broadcasts. In *MTT*, pages 1–3, 2012.

[13] H. Jayakumar et al. Hypnos: An Ultra-low Power Sleep Mode with SRAM Data Retention for Embedded Microcontrollers. In *CODES*, pages 11:1–11:10, 2014.

[14] Huifang Qin. *Deep Sub-Micron SRAM Design for Ultra-Low Leakage Standby Operation*. PhD thesis, University of California, Berkeley, 2007.

[15] J. Kulkarni et al. Process Variation Tolerant SRAM Array for Ultra Low Voltage Applications. In *DAC*, pages 108–113, 2008.

[16] Benjamin Ransford. *Transiently Powered Computers*. PhD thesis, University of Massachusetts Amherst, January 2013.

[17] H. Nakamura et al. Normally-off computing project: Challenges and opportunities. In *ASP-DAC*, pages 1–5, 2014.

[18] B.Ransford et al. Mementos: System Support for Long-running Computation on RFID-scale Devices. *SIGARCH Comput. Archit. News*, 39(1):159–170, 2011.

[19] H. Jayakumar et al. QuickRecall: A HW/SW Approach for Computing Across Power Cycles in Transiently Powered Computers. *J. Emerg. Technol. Comput. Syst.*, 12(1):8:1–8:19, 2015.

[20] H. Jayakumar et al. QUICKRECALL: A Low Overhead HW/SW Approach for Enabling Computations Across Power Cycles in Transiently Powered Computers. In *VLSID*, pages 330–335, 2014.

[21] D. Balsamo et al. Hibernus: Sustaining Computation During Intermittent Supply for Energy-Harvesting Systems. *Embedded Systems Letters, IEEE*, 7(1):15–18, 2015.

[22] A. Rodriguez et al. Approaches to Transient Computing for Energy Harvesting Systems: A Quantitative Evaluation. 2015.

[23] M. Zwerg et al. An 82 uA/MHz microcontroller with embedded FeRAM for energy-harvesting applications. In *ISSCC*, pages 334–336, 2011.

[24] H. Jayakumar et al. Energy-Aware Memory Mapping for Hybrid FRAM-SRAM MCUs in IoT Edge Devices. In *VLSID*, 2016.

[25] W. Yu et al. A non-volatile microcontroller with integrated floating-gate transistors. In *DSN-W*, pages 75–80, 2011.

[26] S. Khanna et al. An FRAM-Based Nonvolatile Logic MCU SoC Exhibiting 100% Digital State Retention at $VDD$= 0 V Achieving Zero Leakage With $<$ 400-ns Wakeup Time for ULP Applications. *IEEE JSSC*, PP(99):1–12, 2013.

[27] Y. Wang et al. A 3us wake-up time nonvolatile processor based on ferroelectric flip-flops. In *ESSCIRC*, pages 149–152, 2012.

[28] N. Sakimura et al. 10.5 A 90nm 20MHz fully nonvolatile microcontroller for standby-power-critical applications. In *ISSCC*, pages 184–185, 2014.

[29] N. Onizawa et al. A sudden power-outage resilient nonvolatile microprocessor for immediate system recovery. In *NANOARCH*, pages 39–44, 2015.

[30] C. Wang et al. Storage-less and converter-less maximum power point tracking of photovoltaic cells for a nonvolatile microprocessor. In *ASP-DAC*, pages 379–384, 2014.

[31] K. Ma et al. Architecture exploration for ambient energy harvesting nonvolatile processors. In *HPCA*, pages 526–537, 2015.

[32] F.A. Aouda et al. Incremental checkpointing of program state to NVRAM for transiently-powered systems. In *ReCoSoC*, pages 1–4, 2014.

[33] B. Lucia and B. Ransford. A simpler, safer programming and execution model for intermittent systems. *SIGPLAN Not.*, pages 575–585, 2015.

[34] A.B. Kahng and S. Kang. Accuracy-configurable adder for approximate arithmetic designs. In *DAC*, pages 820–825, 2012.

[35] V. Gupta et al. Impact: Imprecise adders for low-power approximate computing. In *ISLPED*, pages 409 –414, 2011.

[36] A. Ranjan et al. ASLAN: Synthesis of approximate sequential circuits. In *DATE*, pages 364:1–6, 2014.

[37] D. Shin and S.K. Gupta. A new circuit simplification method for error tolerant applications. In *DATE*, pages 1 –6, 2011.

[38] S. Venkataramani et al. SALSA: Systematic logic synthesis of approximate circuits. In *DAC*, pages 796–801, 2012.

[39] P. Kulkarni et al. Trading accuracy for power with an underdesigned multiplier architecture. In *VLSID*, pages 346 –351, 2011.

[40] A. Lingamneni et al. Energy parsimonious circuit design through probabilistic pruning. In *DATE*, pages 1 –6, 2011.

[41] V.K. Chippa et al. Energy-efficient recognition and mining processor using scalable effort design. In *CICC*, pages 1–4, 2013.

[42] S. Venkataramani et al. Quality programmable vector processors for approximate computing. In *MICRO*, pages 1–12, 2013.

[43] H. Esmaeilzadeh et al. Architecture support for disciplined approximate programming. In *ASPLOS*, pages 301–312, 2012.

[44] S. Sidiroglou-Douskos et al. Managing performance vs. accuracy tradeoffs with loop perforation. In *ESEC/FSE*, pages 124–134, 2011.

[45] S.T. Chakradhar and A. Raghunathan. Best-effort computing: Rethinking parallel software and hardware. In *DAC*, pages 865–870, 2010.

[46] H. Esmaeilzadeh et al. Neural acceleration for general-purpose approximate programs. In *MICRO*, pages 449–460, 2012.

[47] A. Raha et al. Input-based dynamic reconfiguration of approximate arithmetic units for video encoding. *TVLSI*, pages 1–1, 2015.

[48] A. Raha et al. Quality Configurable Reduce-and-rank for Energy Efficient Approximate Computing. In *DATE*, pages 665–670, 2015.

[49] A. Sampson et al. Approximate Storage in Solid-state Memories. In *MICRO*, pages 25–36, 2013.

[50] S. Liu et al. Flikker: Saving DRAM Refresh-power Through Critical Data Partitioning. In *ASPLOS*, pages 213–224, 2011.

[51] A. Raha et al. Quality-aware Data Allocation in Approximate DRAM. In *CASES*, pages 89–98, 2015.

[52] M. Shoushtari et al. Exploiting Partially-Forgetful Memories for Approximate Computing. *Embedded Systems Letters, IEEE*, 7(1):19–22, 2015.