

EXCERPTS FROM

**DEWDROP: AN ENERGY-AWARE
RUNTIME FOR
COMPUTATIONAL RFID**

Michael Buettnner (UW), Benjamin Greenstein
(Intel Labs, Seattle), David Wetherall (UW)

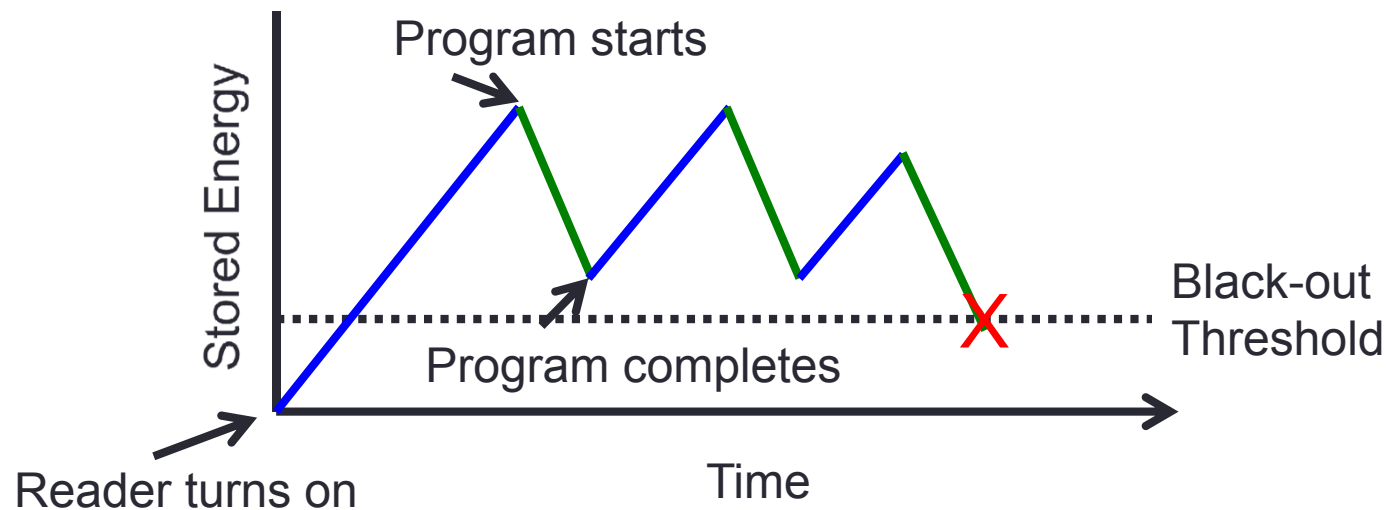




Challenges to Running Programs Efficiently

1. CRFIDs have miniscule energy stores
2. Programs have different energy needs
3. Platform inefficiencies
4. Energy is harvested even while executing

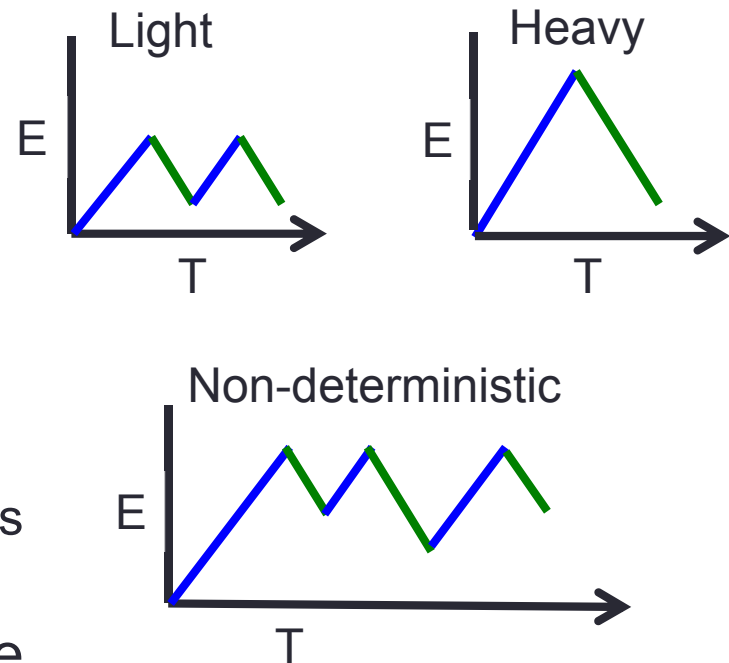
CRFIDs have miniscule energy stores



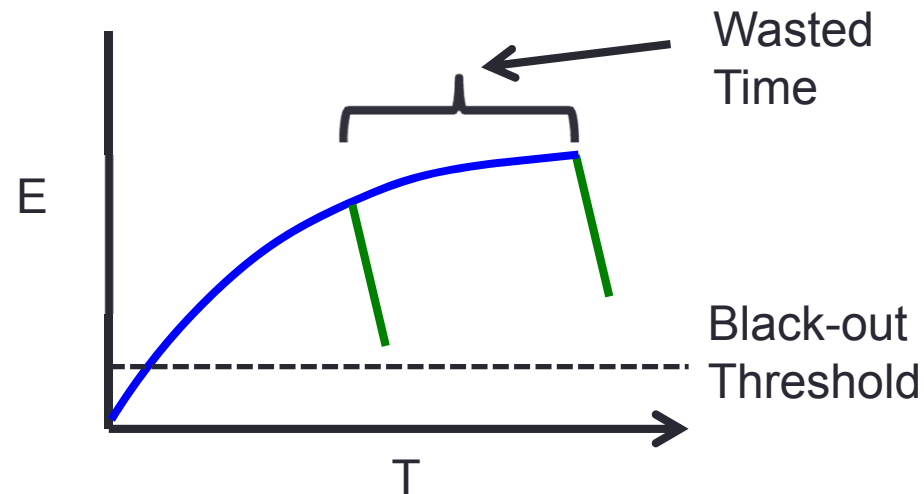
- **Low power mode** ($\sim 1\mu\text{A}$) to store energy, maintain state
- **Active mode** ($\sim 100\text{s of } \mu\text{A}$) to compute and sense
- 100s of ms to charge, 10s of ms to discharge
- **Tags must store enough energy to complete program before beginning execution**

Programs have different energy needs

- Wide range of energy needs
 - E.g., Sense, sense and communicate
- May be non-deterministic
 - E.g., RFID MAC protocol
- Run-to-completion
 - E.g., communication, sampling sensors
- Tags run only one program at a time
- ***Tags must store different amounts of energy when running different programs***

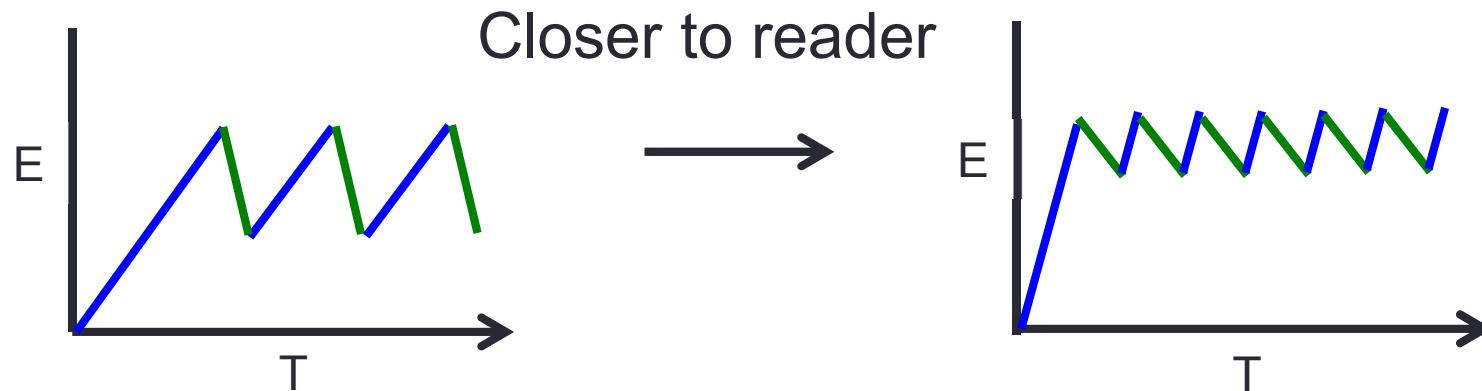


CRFIDs have inefficiencies



- The more stored energy, the longer it takes to store additional energy
 - CRFIDs use capacitors as they are small and can recharge indefinitely
- Voltage regulation → inefficient to operate with more stored energy
- ***Storing excess energy is inefficient***

Energy is harvested even while executing

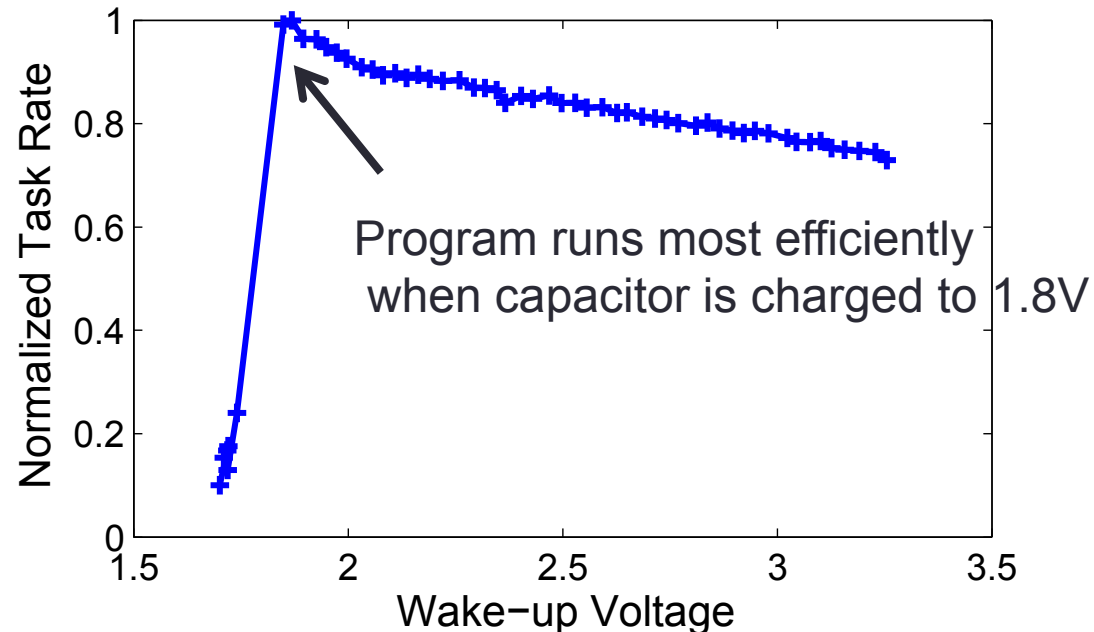


- Received power supplements stored energy
 - Reader frequency hopping → power changes every 400 ms
- ***The amount of stored energy required depends on the distance from the reader and RF environment***



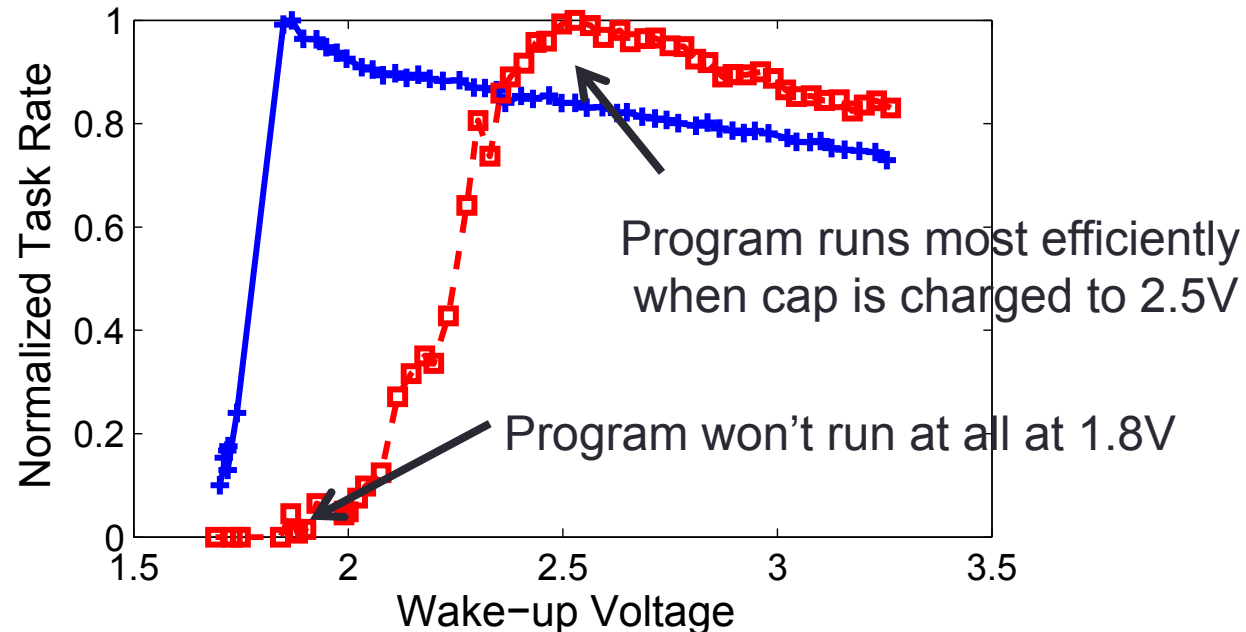
Challenges to Running Programs Efficiently: **Implications**

Storing the right amount of energy increases performance



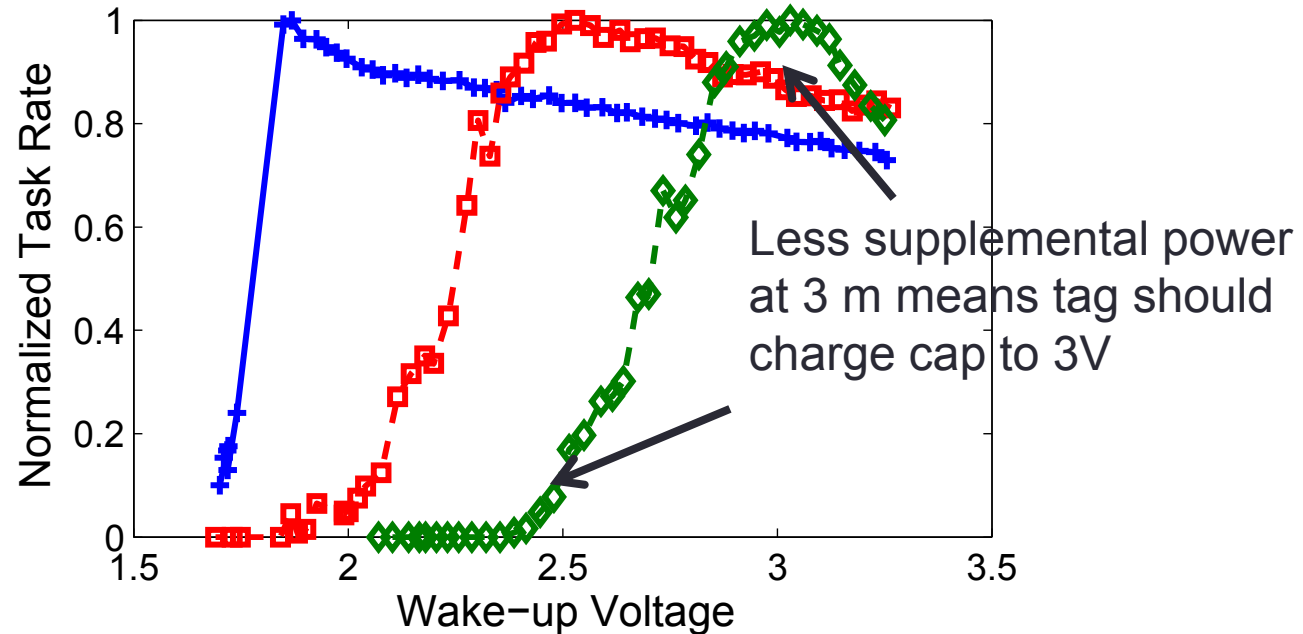
- Wake-up voltage: Determines the amount of energy stored before starting program
- Light WISP program: sample accelerometer, 1.5 m from reader

No fixed threshold works for all programs



- Heavy, non-deterministic program: sample accelerometer and transmit value to reader, 1.5 m

No fixed threshold works for all distances



- Heavy, non-deterministic program, **3 m from reader**
- **CRFID must adapt to program needs and environment**



Outline

- Intel WISP – A CRFID Tag
- Challenges to Running Programs Efficiently
- Dewdrop Design
- System Evaluation

Dewdrop: An Energy-aware Runtime

Adaptively find the **wake-up voltage** that **maximizes execution rate** for the **program** and **RF environment**

Two factors that reduce execution rate:

- Not storing enough energy: Program fails and it takes time to recharge and execute again
- Storing too much energy: Overcharging wastes time

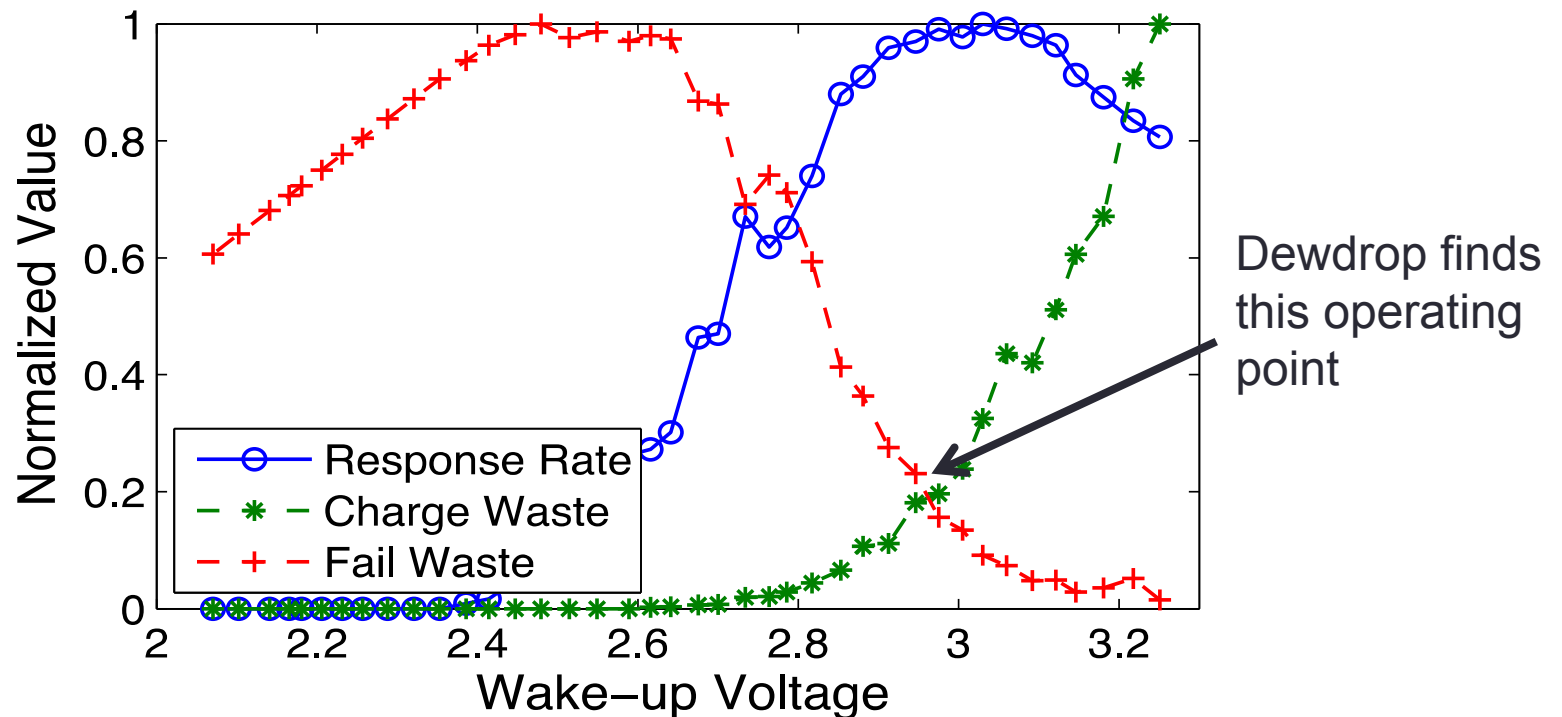
Constraint: Runtime operation must be simple

- Every active cycle costs energy
- No floating point, no hardware multiply/divide

Adapt to the Program and Environment

- **Goal:** Maximize execution rate →
Minimize time wasted from program failure and overcharging
- **Heuristic:** Total waste is minimized when the wasted time from failures and overcharging is equal
- **On program complete:**
Update running average of time wasted overcharging
- **On program failure:**
Update running average of time wasted failing
- **If** $\text{Avg}_{\text{overcharge}} > \text{Avg}_{\text{fail}}$: decrease wake-up threshold by β
- **Else:** increase wake-up threshold by β

Heuristic results in a good operating point



- **Equalizing the sources of wasted time results in efficient program execution**

Dewdrop Implementation

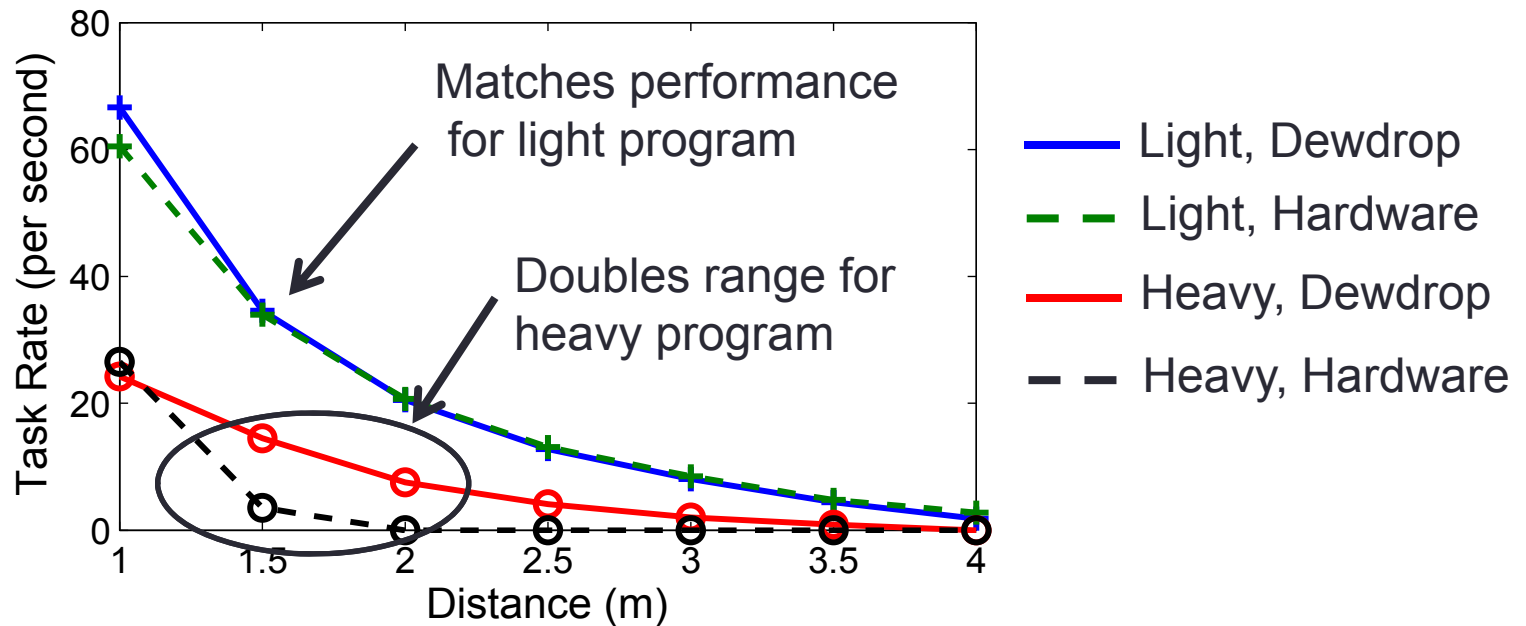
1. Low power wake-up
 - No hardware mechanism to wake up at specified voltage
 - Dewdrop polls capacitor voltage periodically until target is reached
 - Exponentially adapted polling interval is lightweight and accurate
2. Low power voltage sampling
 - Waking up to sample voltage consumes precious energy
 - We reduced the energy cost of voltage sampling by a factor of 4

More details in the paper



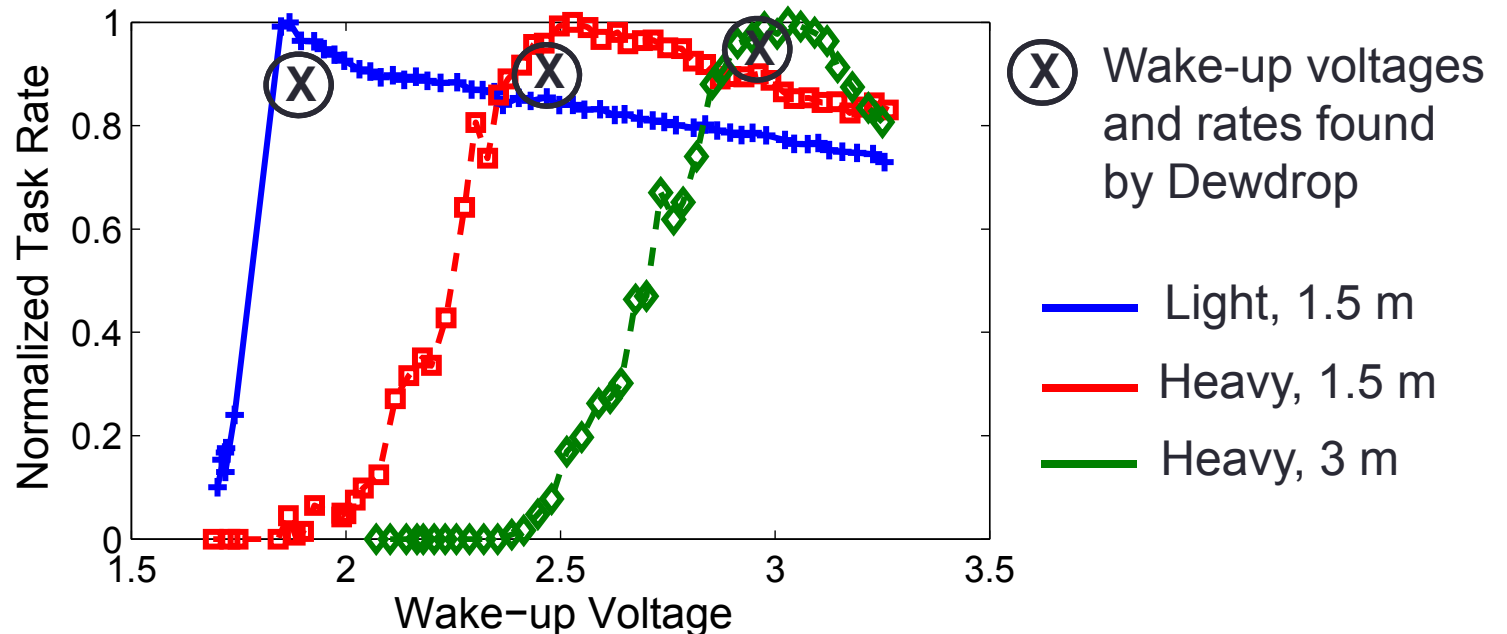
System Evaluation

Dewdrop makes good use of scarce energy



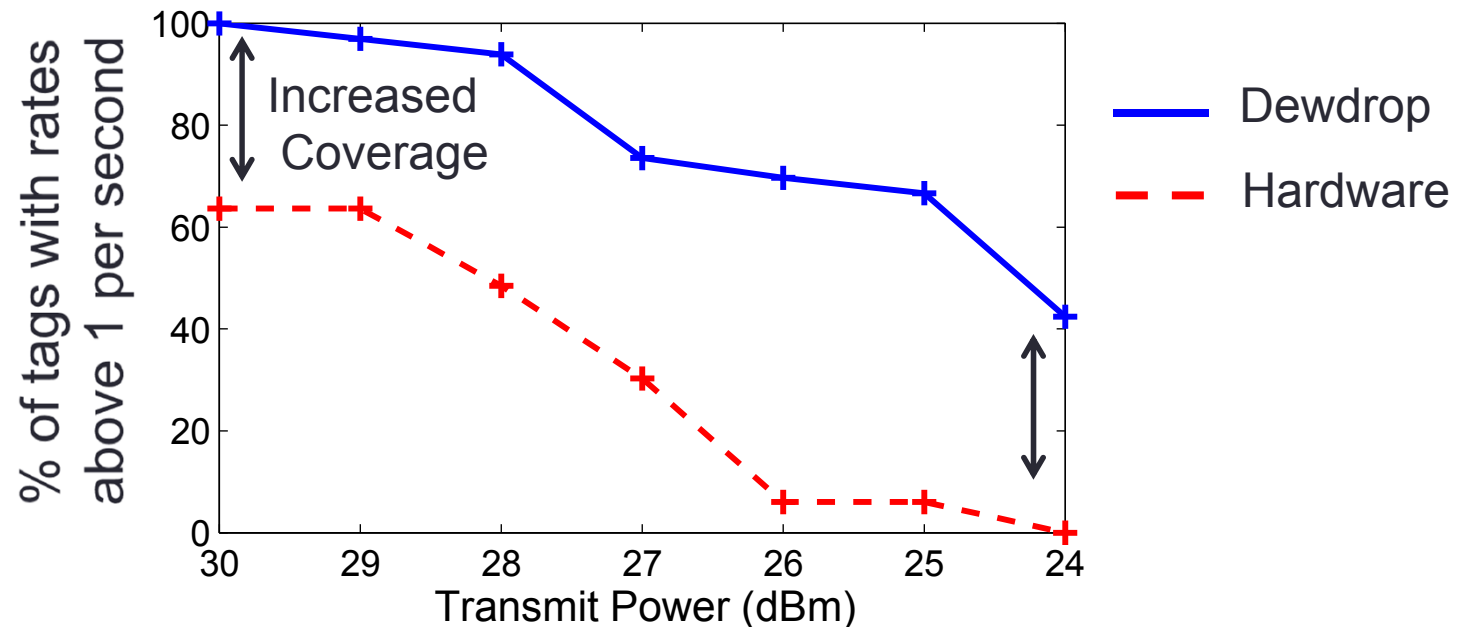
- Compare to efficient, but inflexible, hardware mechanism
 - State-of-the-art before Dewdrop
- Execution rate should scale with received power: $1/d^2$

Dewdrop finds an efficient operating point



- Dewdrop finds wake-up voltage within 0.1V of best
- Generally achieves > 90% of max rate for all distances

Dewdrop increases application coverage



- Elder care scenario: 1 reader, tagged objects in apartment
- 11 WISPs streaming accelerometer data (3 trials)
- **Dewdrop can run the program with much less power**

Conclusion

- Running programs using harvested RF energy is feasible
 - Batteryfree → small, perpetual, embeddable
- Dewdrop makes CRFIDs more usable and useful
- Technology trends will increase range and performance
 - Passive device range expected to continue doubling every 4 years
 - WISP 5.0 in development
- WISPs and tools are available to the community
 - WISP hw/sw open source, USRP-based RFID reader

Questions

- WISP Wiki: wisp.wikispaces.com
- UW Sensor Systems Group: sensor.cs.washington.edu
- www.cs.washington.edu/homes/buettner
- buettner@cs.washington.edu