Energy-Adaptive Real-time Sensing for Batteryless Devices

Mohsen Karimi, Yidi Wang, and Hyoseung Kim

University of California, Riverside
Batteryless Devices

• **Definition:** Devices that do not use battery and are usually powered by intermittent power sources such as sunlight, heat, vibration, and radio signals.

• **Applications:** Smart homes, agriculture, health monitoring, …

• **Advantages:**
  – Few maintenance is required
  – They can last for decades
  – They can be deployed in extreme environments

• **Examples:** Beacon, R’tag, Flicker, ...
Previous work on Task Scheduling on Batteryless Devices

Static methods:

• Target periodic task execution with known charging behavior and try to meet deadlines for real-time tasks[1,2]

Reactive methods:

• Try to schedule tasks of a given taskset when the system is provided with an assumed amount of energy supply [3,4]

Limitations of Previous work

• Unreliable energy source
  – The energy pattern needs to be known a priori
  – The schedule does not adapt to the power reception rate

• In many sensing applications, **data freshness** of tasks is often more important than increasing the number of executed jobs or reducing the number deadline misses.
  – Example: Health monitoring systems
    • Blood sugar monitoring system
Proposed Framework

- Energy harvester
- Energy storage and regulator units
- Voltage and harvesting rate monitoring
- Energy Predictor
- Scheduler
  - LASF
    - Job Skipping
    - Elastic Period
  - EDF
    - Job Skipping
    - Elastic Period
Data Freshness and Age of Information (AoI)

- AoI of a task $\tau_i$ at time $t$, $A_i(t)$, is the time elapsed since the latest output of the task was generated.

\[
\mu_{A_1} = \frac{\int_{t_1}^{t_4} A_1(t) \, dt}{t_4 - t_1} = \frac{\sum_{i=1}^{3} (t_{i+1} - t_i)^2}{2 \times (t_4 - t_1)}
\]
Task Model

• Task model
  • $C_i$: Execution time
  • $T_i$: Period
  • $D_i$: Relative deadline
  • $MTA_i$: Maximum tolerable age of information
  • Tasks are non preemptive

\[
\tau_i = (C_i, T_i, D_i, MTA_i)
\]

\[
\forall i \leq n \mid MTA_i \geq T_i \land D_i \leq T_i.
\]

• Charging model [1]
  • $m_a$: Charging rate of the system
  • $m_{Pi}$: Discharging rate of the task $\tau_i$
  • $Q_i$: Charging time required before starting the task $\tau_i$

\[
Q_i = \frac{(m_{Pi} - m_a) \times C_i}{m_a}
\]

Task Scheduling with energy constraint

- Earliest Deadline First (EDF) scheduling with energy constraint

\[ Q_i^+ = \max\{Q_i, 0\} \]

Blocking time: \[ B_k = \max_{j : D_i > D_k} C_j \]

\[ \forall i, j \leq n, i \leq j \rightarrow D_i \leq D_j \]

Schedulable if:

\[ \forall k = 1, \ldots, n, \quad \sum_{i=1}^{k} \left( \frac{C_i + Q_i^+}{D_i} \right) + \frac{B_k}{D_k} \leq 1 \]

- Relative deadlines no larger than period
  \[ \forall i = 1, \ldots, n, \quad D_i \leq T_i \]

- Unpredictable behavior in overload or underload situation (decrease or increase in charging rate)
Task Scheduling with energy constraint

- Earliest Deadline First (EDF) scheduling with energy constraint and overload management
  - Elastic Period
  - Job skipping: Skip jobs if the previous job of the same task has not finished its execution

\[ U_e = \max_{k \leq n} \left\{ \sum_{i=1}^{k} \left( \frac{C_i + Q_i^+}{T_i} \right) + \frac{B_k}{T_k} \right\} \]

\[ T_i' = T_i \times U_e \]

\( \tau_1: C=1, T=2, D=2 \)
\( \tau_2: C=1.5, T=3.5, D=3.5 \)
\( \tau_3: C=2.5, T=6, D=6 \)
LASF Task Scheduler

- Least AoI Slack First (LASF)
  \[ U_l = \max_{k \leq n} \left\{ \sum_{i=1}^{k} \left( \frac{C_i + Q_i^+}{MTA_i} \right) + \frac{B_k}{MTA_k} \right\} \]

  - Job Skipping
    - Similar to EDF

  - Elastic Period
    - \( T'_i = \hat{MTA}_i \)
    - This guarantees each task \( \tau_i \) to meet its AoI constraint of \( \hat{MTA}_i \)

Algorithm 1 Least AoI Slack First

1: \( t \leftarrow \) current time
2: Update \( A_i \) of each task
3: Compute \( U_l \) by (7)
4: for \( k \leq n \) do
5: \( \hat{MTA}_k = U_l \times MTA_k \) \( \triangleright U_l \) is obtained by (7)
6: \( ASD_k = \hat{MTA}_k - C_k \)
7: end for
8: \( h \leftarrow \arg \min_i (ASD_i - A_i) \)
9: if \( Curr\_Charge \geq Q_i \) then
10: Execute the task \( \tau_i \)
11: else
12: \( t_r \leftarrow \) earliest release time of a job from any of the tasks
13: \( t_c \leftarrow t + Q_i - Curr\_Charge \)
14: \( t_{new} \leftarrow \min\{t_c, t_r\} \)
15: \( t \leftarrow t_{new} \) \( \triangleright \) Device goes to sleep for \( t_{new} - t \) seconds
16: end if
Considerations for Energy Prediction

• Requirements
  – Lightweight due to limited processing capability
  – Relatively accurate
  – No additional sensor information needed

• Previous work
  – EWMA [5] - baseline
  – WCMA [6]

Energy Predictor

• Neural Network Based Predictor
  – The solar radiation of each time slot is relatively correlated to the past few hours of that time slot as well as the same time slot for past few days
  
  – Inputs (for 30 minutes time slot):
    • 10 samples from previous time slots (past 5 hours)
    • 4 samples from 4 previous days
  – 12 neurons in hidden layer
  – NREL data set[8]
    • Trained with data from 2017 to 2019

Evaluation: Scheduler Performance

- **Objective:** Explore effect of utilization
  - 1000 tasksets
  - Utilization from 0.05 to 0.9 in 0.05 steps
  - $m_a = 3$
  - UUniFast [9] is used to generate random tasksets
  - Discharging rate of each task are chosen randomly from 2 to 10
  - MTA for each task is chosen randomly from 1x to 4x of its period

Evaluation: Scheduler Performance

- **Objective: Explore effect of task energy demand**
  - Energy demand task ratio effect
    - High demand tasks: discharging from 8 to 10
    - Low demand tasks: discharging from 1 to 3
    - 1000 tasksets
    - $m_a = 3$
    - UUniFast is used to generate random tasksets
    - MTA for each task is chosen randomly from 1x to 4x of its period

- **For elastic period adjustment**, LASF and EDF are similar
- **For job skipping**, LASF is better than EDF

**Graph:**
- AoI improves with lower energy demand
- For elastic period adjustment, LASF and EDF are similar
- For job skipping, LASF is better than EDF
Evaluation: Energy Predictor

- **Objective: Prediction accuracy**
  - Test set: data for the entire year of 2020
    - NREL data set
    - Evaluation is performed on Raspberry Pi 3

![Mean Absolute Error (W/m²)](chart)

*For reference: average solar energy rate is about 800 to 1200 Watt/m² at noon*
Evaluation: Energy Predictor

• **Objective: Runtime overhead**
  - Test set: data for the entire year of 2020
    - NREL data set
    - Evaluation is performed on Raspberry Pi 3

![Average execution time (ms) graph](image-url)
Conclusion and Future Work

• We presented a new framework for scheduling real-time sensing tasks with data freshness constraints on batteryless devices.

• We proposed a lightweight machine learning based solar energy predictor

• Our proposed predictor outperformed the state-of-the-art methods in terms of mean absolute error as well as runtime overhead

• We studied job skipping and elastic period adjustment methods to deal with overload situations

• The combination of tasks with AoI constraints and hard deadlines can be considered
  – Preprocessing task can be performed before sensing tasks
  – Transmission tasks over low power medium such as BLE can be scheduled after each sensing

• We plan to explore these issues and further evaluate our methods in real-world conditions
Thank You