# Invited paper: An Efficient Energy Management Solution for Renewable Energy Based IoT Devices

Elizabeth Liri[1], K. K. Ramakrishnan[1], Koushik Kar[2], Geoff Lyon[3] and Puneet Sharma[3]
[1] University of California Riverside, USA [2]Rensselaer Polytechnic Institute, USA [3]Hewlett Packard Labs, USA

## ABSTRACT

Multi-sensor IoT devices enable the monitoring of different phenomena using a single device. Often deployed over large areas, these devices have to depend on batteries and renewable energy sources for power. Therefore, efficient energy management solutions that maximize device lifetime and information utility are important. We present SEMA, a smart energy management solution for IoT applications that uses a Model Predictive Control (MPC) approach to optimize IoT energy use and maximize information utility by dynamically determining task values to be used by the IoT device's sensors. Our solution uses the current device battery state, predicted available solar energy over the short-term, and task energy and utility models to meet the device energy goals while providing sufficient monitoring data to the IoT applications.

To avoid the need for executing the MPC optimization at a centralized sink (from which the task values are downloaded to the SEMA devices), we propose SEMA-Approximation (SEMA-A), which uses an efficient MPC Approximation that is simple enough to be run on the IoT device itself. SEMA-A decomposes the MPC optimization problem into two levels: an energy allocation problem across the time epochs, and task-dependent sensor scheduling problem, and finds efficient algorithms for solving both problems. Experimental results show that SEMA is able to adapt the task values based on the available energy, and that SEMA-A closely approximates SEMA in sensing performance.

## CCS CONCEPTS

• **Networks** → **Network experimentation**; *Network measurement*; **Network performance analysis**; • **Computer systems organization** → **Sensor networks**.

## KEYWORDS

IoT, energy efficiency, optimization, renewable energy, MPC

**ACM Reference Format:**
Elizabeth Liri[1], K. K. Ramakrishnan[1], Koushik Kar[2], Geoff Lyon[3] and Puneet Sharma[3]. 2023. Invited paper: An Efficient Energy Management Solution for Renewable Energy Based IoT Devices. In *24th International Conference on Distributed Computing and Networking (ICDCN 2023), January 4–7, 2023, Kharagpur, India.* ACM, New York, NY, USA, 8 pages. https://doi.org/10.1145/3571306.3571387

## 1 INTRODUCTION

Multi-sensor IoT devices can be used to monitor different phenomena. Often deployed over large areas, these devices depend on batteries and renewable energy sources for power. Therefore, efficient energy management solutions are needed to maximize device lifetime while ensuring sufficient data is sent to the IoT application.

Various methods can be used to manage device energy efficiently including predicting renewable energy availability [11], adjusting task schedules [4], and adjusting workloads [12]. Examples of IoT deployments in the field include Signpost [1], a platform for city-scale sensing; FarmBeats [20], an IoT platform for data-driven agriculture and [16] a solution for monitoring soil moisture. SEMA uses task adaptation to maximize device operation throughout a 24-hour period, despite variability in the availability of renewable energy. Compared to [16], SEMA uses a less complex MPC-based approach. Further, SEMA-A can be implemented at the IoT device at low complexity, unlike [20] which has a centralized solution. Furthermore, SEMA considers the task energy requirements when optimizing the task values, unlike [1] which can leave some tasks energy starved while other tasks have excess energy.

This work presents SEMA, an adaptive, efficient energy management solution for IoT. It is based on the SEMA Stick hardware that we have developed, which is a portable multi-sensor platform that includes a low-power Raspberry Pi for computation. It also contains a small solar panel, a low-cost battery, and multiple sensors. Our SEMA algorithm uses the current battery state, predicted available solar energy, and task energy and utility models in an MPC-based algorithm to dynamically determine appropriate task values that maximize information utility while ensuring device energy goals are met. We divide the day into 15 min epochs and run our SEMA algorithm every epoch.

Since our MPC-based approach has to run on a centralized sink, and then have task values downloaded to the IoT device, it is not always practical for deployment. In view of this, we propose SEMA-Approximation (SEMA-A), a modified and efficient MPC Approximation approach that is simple enough to be run on the IoT device itself. SEMA-A decomposes the MPC optimization problem into two levels: an outer-level energy allocation problem, and an inner-level sensor scheduling problem. The energy allocation problem fairly splits the total available energy for the rest of the day into the remaining epochs and the sensor scheduling problem divides the epoch energy between all tasks. By exploiting the specific problem structure, we show that the outer-level energy allocation problem can be solved independently of the inner-level sensor scheduling problem, and at low complexity. Once the former is solved, the latter can be easily solved to obtain the optimal sensor task values. Due to space limitations, we present a high-level discussion of the proof, with the full proof and additional information in [14]. Experimental results show that SEMA adapts the task values effectively based on the available energy. The results also demonstrate that SEMA and SEMA-A have very similar performance.

## 2 BACKGROUND AND RELATED WORK

Examples of IoT deployments include Farm Beats [20], an IoT solution for agriculture that uses sensors and drones to improve farm productivity. Their solution uses a base station with a weather-aware solar prediction algorithm and adjusts the duty cycle of different base station components to save energy. LoRaWAN based IoT networks are used in [5] and [19]. [5] focuses more on IoT service security, while [19] uses multi-sensor low-cost IoT devices to improve productivity in a vineyard.

[16] predicts the hourly soil moisture content to improve farm irrigation using a neural network approach. They use eleven distinct soil and environmental parameters and compare prediction results from two optimization techniques: Scaled Conjugate Gradient and BFGS Quasi-Newton. They then use the best optimization technique for predicting the hourly variation in soil moisture. The prediction results are forwarded to a fuzzy logic based weather model that determines the appropriate irrigation schedule depending on the weather at that particular location.

Signpost [1] is an example of a city-scale sensing platform with similar components as SEMA i.e. Signpost includes a solar prediction component, independent modular sensor components, and an energy management algorithm. Other examples of deployments include improving mushroom farming with Zigbee [13] and PotatoNet [9], which describes the experiences and challenges of an outdoor deployment.

There are many approaches for IoT energy management that adapt task values or workloads. [4] uses dynamic programming and schedules tasks based on energy availability. In their approach each task has multiple versions with different performance levels (QoS) and only one task version is selected for execution each epoch. Using the constraint of energy neutrality, their algorithm finds the task schedule that maximizes QoS. The solution space of possible solutions in their dynamic programming approach is reduced due to the number of time slots to be scheduled and the set of battery levels. Thus they have a pseudo polynomial time complexity of $O(KB_{max})$ where $K$ is the number of time slots and $B_{max}$ is the maximum battery level. They also propose a solar prediction algorithm that assumes the solar energy during the day follows a parabolic curve with no solar at the beginning and end of the day and peak solar at midday. They obtain historical daily total solar irradiance data for a geographical location from the NASA program RETscreen and use the zenith angle at each hour to determine the hourly solar irradiance.

[12] focuses on energy-neutral operations and recommends workloads be selected according to the current energy availability but do not implement their approach. In [11], the authors monitor path loss to determine efficient transmission power levels and using a predictive power management approach they then match the system power consumption to the average energy generation rate.

SEMA uses a simpler, less expensive MPC-based approach compared to [4] and [16]. It also adapts the workloads to the available energy similar to [12], [4] and [11], but at individual task level rather than at the overall duty-cycle level like [12]. The SEMA solar prediction adapts every epoch and is unique to an individual IoT device based on its battery state while [4] and [1] use the same solar predictions for all devices in the same geographical area. The SEMA solar prediction also does not require a lot of history or require weather data unlike [16]. It uses low-cost COTS devices and does not have a centralized design like [20]. Unlike [11] that adapts power transmission levels, SEMA adapts the task sensing parameters.

Model Predictive Control (MPC) [7] has been used to predict the behavior of a system over a finite time window using a system model. MPC is an advanced process control method that can handle systems with multiple inputs and multiple outputs with various interactions between the inputs and outputs. To drive the predicted output e.g. predicted battery state towards a reference value e.g. battery value at a given deadline, MPC solves an online optimization algorithm. The MPC optimization thus finds the optimal control action based on the dynamic system model and constraints that will allow the system to achieve the desired output. In an IoT context, MPC has been used for energy management to optimize power consumption and indoor thermal comfort. For example, MPC has been used for optimizing operations of indoor heating and cooling [8], Heating, Ventilation and Air Conditioning (HVAC) [3] and smart wireless systems [17]. In SEMA we use MPC to optimize IoT device power consumption in an outdoor environment unlike [3] and [8] which operate in an indoor environment. In the solutions mentioned, MPC is executed in a centralized server. However, with SEMA we propose an MPC approximation approach that can be run autonomously on constrained IoT devices.

## 3 SYSTEM ARCHITECTURE

### 3.1 Overview

SEMA balances two competing objectives: limiting energy use to extend the device lifetime, and expending energy executing sensing tasks to satisfy the IoT application's information requirements. We use four types of tasks: the video task which uploads one streaming video per epoch and varies the video duration $t_v$; the image task uploads one image per epoch and varies the quality $q$; the temperature and humidity tasks vary the number of measurements per epoch, $n_t$ and $n_h$, respectively.

Every epoch (15 mins), the SEMA algorithm determines the available energy for use at the IoT device. It does this by first measuring the current battery state to determine how much battery energy is left. Then it uses our solar prediction algorithm to determine the predicted solar energy available from the current epoch till the end of the prediction horizon (06:00 the next morning). The IoT device energy goal is to ensure that there is at least minimum energy remaining in the device at the end of the prediction horizon, i.e., the device does not shut down due to low energy before 06:00 the next morning. Therefore, considering the total available energy for use, the energy cost for each task, and the task information utility, SEMA uses the MPC algorithm to determine the appropriate task parameters to be used for the rest of the day that maximize information utility while meeting the energy goal.

The SEMA MPC approach is used during the day and at night a simple night-time algorithm is used since no solar energy is available. The night-time algorithm which is also executed every epoch basically divides the total energy available overnight equally between the epochs and allocates the per epoch energy to the tasks using a weighted max-min approach. Since the algorithms are executed every epoch, they adjust the task values quickly to any solar prediction errors, weather changes, or unanticipated battery changes. When the MPC optimization result is infeasible or the energy per epoch in the night-time algorithm is insufficient to run tasks even at their minimum values, the minimum task parameters are used for that epoch.

Due to its complexity, the SEMA MPC algorithm is executed at a centralized node. Then, task values are downloaded onto the IoT device. However, this is not feasible for all deployments. Thus, we propose SEMA-Approximation (SEMA-A) which uses an MPC approximation algorithm that is simple enough to run on the IoT device itself. SEMA-A gives results close to the exact MPC approach.

## 3.2 SEMA Applications

The SEMA solution can be used in many application areas since it balances energy use and information utility. Examples of application areas include agriculture, wildfire monitoring, and environment monitoring in urban areas.

In agriculture, IoT devices can be used to monitor soil, environment, and crop conditions in a field. This data is then used for precision agriculture allowing farmers to provide a precise and appropriate response to any variability in the soil or crops. Using SEMA with multi-sensor IoT devices ensures that the devices remain operational overnight while monitoring data is still sent to the sink.

In the case of wildfire monitoring, SEMA can be used to monitor large areas and prolong device lifetime while providing sufficient data to ensure wildfires are detected early. Some existing solutions like Alert Wildfire! [21] and High-Performance Wireless Research and Education Network (HPWREN) [6] use cameras mounted on wide area wireless network base stations to monitor for wildfires. These solutions are expensive to install and have to monitor very large areas. However, if the fire is far away from the base station camera, it may need to reach a significant size before it can be detected. The SEMA solution can be deployed on smaller IoT devices that can monitor smaller areas and forward data to the base station enabling fire detection earlier.

IoT devices used for monitoring the environment in urban areas may still use batteries and have renewable energy sources instead of relying on brown energy. SEMA can be a possible solution to manage device energy and information utility goals. Like Signpost [1], an IoT device running SEMA can have multiple independent sensors, use them to monitor various environmental parameters, and forward the data to a sink which can then provide the necessary targeted response.

The key features of SEMA are that it can balance energy use with the information goals of the IoT device. This makes the SEMA energy management solution very flexible and practical and can be used in different application scenarios.

## 3.3 SEMA Hardware Design

Fig 1 shows the SEMA Stick, a portable computational unit designed to work with the SEMA software. The hardware consists of a solar panel, a low-cost Lithium-ion battery, Raspberry Pi Zero W (RPZ), and a microcontroller. The SEMA stick has an auto-sensing switch that automatically switches between solar and battery energy sources. If the solar energy is too low to support the load (RPZ) then it switches to the battery as the energy source. The microcontroller contains multiple independent sensors such as temperature, humidity, and soil moisture sensors and the RPZ includes a camera to provide video and image data. The SEMA Stick is designed to work with the SEMA algorithm to modify the sensor operations in response to changes in the available energy.

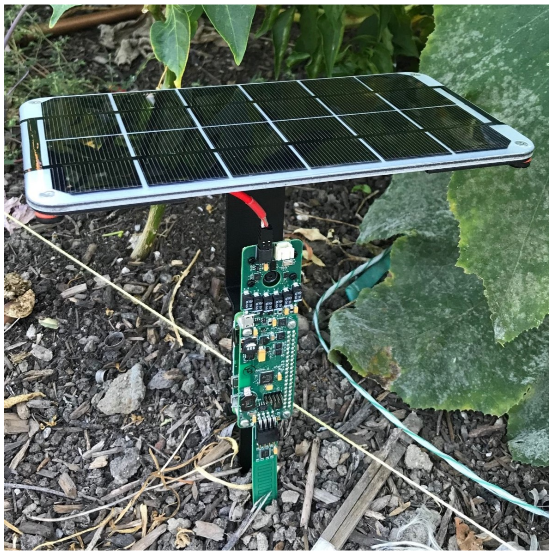
**Figure 1: SEMA Stick Hardware Unit.**

Due to development costs, there are two constraints with the prototype design. First, the device cannot enter/exit low-power sleep states on demand, therefore, the device is always powered on. Second, since the device has a solar power component, the maximum energy that can be captured at the solar panel is the sum of the current energy required by the battery (for recharging) and the load (for running the RPZ and sensors). These two constraints have been taken into account in our design.

## 3.4 SEMA Algorithm Design

*3.4.1 MPC Framework.* SEMA uses a Model Predictive Control (MPC) approach [7] to optimize the task values during the day, while ensuring that the battery energy remains sufficient for the sensor to remain operational till the next morning without shutting down. SEMA runs every epoch and updates the task values based on the current battery state and solar prediction for the rest of the day; therefore it quickly adapts to changes in the environment. Let each task $k \in \mathcal{K}$ be associated with a utility function $U_k$ and weight $w_k$ that captures the importance of the task compared to the others. Each task also has an energy cost $E_k$, and both $E_k$ and $U_k$ are a function of the task's variable parameter represented by $f_k$. The parameter $f_k$ represents video duration $t_v$, image quality $q$, temperature measurement count $n_t$ and humidity measurement count $n_h$, respectively, for the four sensing tasks. Then the optimization run by MPC in every epoch can be represented as follows.

$$\max_{f_k(n)\geq f_{k,min}} \sum_{n=n_0}^{N} \sum_{k\in\mathcal{K}} w_k U_k(f_k(n)), \tag{1}$$

subject to

$$E(n) = \min\left[E(n-1) + R(n), E_{max}\right], \ \forall n, \tag{2}$$

$$R(n) = \min\left[S(n) - \sum_{k\in\mathcal{K}} E_k(f_k(n)) - E_{base}, R_{max}(n)\right], \forall n, \tag{3}$$

$$R_{max}(n) = \frac{IeE(n)v_{max}}{E_{max}}, \ \forall n, \tag{4}$$

$$E(n) \geq E_{min}, \ \forall n. \tag{5}$$

In the above equations, $E(n)$ is the battery state at the beginning of epoch $n$, and $E_{max}$ ($E_{min}$) is the maximum (minimum) battery energy level allowed at any time. The total energy needed by task $k$ in epoch $n$ is represented by $E_k(f_k(n))$, and $S(n)$ is the total predicted solar energy available for use during epoch $n$. Further, $R(n)$ is the energy pushed into or taken out of the battery during epoch $n$, and $R_{max}(n)$ is the recharge limit. Energy used by the SEMA Stick when idle is $E_{base}$. In the expression for $R_{max}$ in Eq. (4), $e$ is the epoch duration (seconds) and $I$ is the current flowing in the battery in epoch $n$. Maximum battery voltage $v_{max}$ is 4.2V and $f_{k,min}$ is the required minimum sensing level for sensing task $k$.

The MPC objective function expressed in Eq. (1) is to maximize the total information utility under the constraints given by Eqs. (2)-(5). Here, Eq. (2) represents the battery energy evolution, while Eqs. (3) and (4) show how the recharge rate $R(n)$ is calculated. In Eq. (3), $S(n) - \sum_k E_k(f_k(n)) - E_{base}$ represents the energy that is available for recharging the battery once the idle energy and energy for sensing tasks are accounted for. If no feasible solution to this MPC is possible then all task values are set to minimum. The first epoch ($n_0$=0) starts at 06:00 am and the last epoch ($N$=95) starts at 05:45 am the next morning.

### 3.4.2 *Low-complexity MPC Approximation.*

The SEMA MPC problem is a complex question that may need to be solved by a convex optimization solver running at a remote sink, from which the computed task values for every epoch are downloaded to each IoT device. However, we would ideally like it run on the IoT device itself – for communication cost (energy) savings and applicability in scenarios where a central server may not be available. Towards that goal, we propose SEMA-Approximation (SEMA-A) which can run on IoT devices (runs in cubic time) and provides performance results similar to the SEMA MPC solution. SEMA-A decomposes the MPC optimization problem of Eqs. (1)-(5) into two levels by carefully approximating some of the constraints in Eqs. (3)-(4).

***Bi-level decomposition:*** Let the vector of all sensing variables over all epochs be represented by $f = (f_k(n), k \in \mathcal{K})$, and let $f \in F$ represent the constraints in Eqs. (2)-(5). Also let $\sum_k E_k(f_k(n)) = g(n)$. Since the sensing variables $f_k(n)$ appear in Eqs. (2)-(5) only in the form $\sum_k E_k(f_k(n))$, we can rewrite the constraint $f \in F$ as $g \in G$ where we obtain $G$ by replacing the term $\sum_k E_k(f_k(n))$ in (3) by $g(n)$. The optimization problem in (1)-(5) can be rewritten as,

$$\max_{g\in G} \sum_{n=n_0}^{N} V(g(n)), \tag{6}$$

$$\text{where } V(g(n)) = \max_{\substack{\sum_k E_k(f_k(n))=g(n),\\ f_k(n)\geq f_{k,min}}} \sum_{k\in\mathcal{K}} w_k U_k(f_k(n)). \tag{7}$$

These two equations represent the bi-level decomposition and split the algorithm into two levels: (i) *Energy allocation per epoch (EA)*, where we solve Eq. (6) and allocate the optimum energy $g^*(n)$ across the epochs $n$; (ii) *Sensor scheduling within an epoch (SS)*, where we solve Eq. (7) for each epoch $n$, dividing up the energy $g^*(n)$ among the different sensing tasks $f_k(n)$ in that epoch. The EA problem is solved once for the entire optimization period but the SS problem is solved for every epoch.

***Sensor scheduling within an epoch (SS):*** The energy equations representing the tasks can be expressed or approximated as $E_k(f_k) = E_k^0 + E_k^1 f_k$ (where constants $E_k^0$ and $E_k^1$ are determined through experimentation). Having the energy equations in this form indicates a linear relationship between the task variable parameter $f_k$ and the task energy cost $E_k$. Using this form enables us to rewrite the original MPC optimization to then solve the sensor scheduling component using a Lagrange multiplier approach. Further, the utility functions have the form $U_k(f_k) = 1 - e^{-\alpha_k f_k}$ where $\alpha_k$ depends upon the sensing task $k$ (see [14] for details). We consider a constraint set with two constraints: lower bounds $f_k(n) \geq f_{k,min}, \forall k$ and a linear constraint $\sum_k E_k^0 + E_k^1 f_k(n) = g(n)$. Algorithm 1 determines the solution that maximizes $\sum_k U_k(f_k(n))$ subject to the constraint $\sum_k E_k^0 + E_k^1 f_k(n) = g(n)$ (see line 5 in Alg. 1) and has a complexity of $O(K^2)$ per epoch, for $K = |\mathcal{K}|$ sensing tasks. If the algorithm finds any resulting $f_k(n) \leq f_{k,min}$ then the $f_k(n)$ value is set to $f_{k,min}$ and we repeat the process until all remaining tasks have $f_k(n) > f_{k,min}$.

The equation in line 5 was determined by using Lagrange multipliers, setting the partial derivatives of the Lagrangian to be zero and solving for $f_k$. The equation for determining $f_k$ for a task is the sum of two parts. Let us refer to $\ln((w_k\alpha_k)/E_k^1)$ as a task-dependent constant and $\sum_{k\in\mathcal{K}'}(E_k^1/\alpha_k)$ as the energy utility constant which is common for all tasks. The second part of the equation finds the extra energy available once the energy required to run all the tasks at minimum values has been subtracted. Using the task-dependent constant, the energy utility constant, and the calculated extra available energy, this part determines the amount to increase or decrease the task parameter value in order to apportion the extra available energy across the different sensors. The correctness of Algorithm 1 in computing the optimum solution in Eq. (7) is shown in [14].

***Energy allocation between epochs (EA):*** The second part of the bi-level decomposition is splitting the available energy between the epochs. We developed an efficient algorithm to solve this problem and proved its correctness. We give an outline below; the full analysis is quite involved and can be found in [14]. The solution relies on two observations. First, while $R_{max}(n)$ varies linearly with $E(n)$ (see Eq. (4)) the slope is nearly flat in the constant charging region [10]. Therefore, we can approximate $R_{max}$ to be a constant, setting it to the lowest value in the constant charging region.

Lemma L.1. *If $R(n) = R_{max}$ $\forall n$, then problem in (6)-(7) is equivalent to maximizing $\sum_{n=n_0}^{N} V(g(n))$, subject to $g(n)$ satisfying*

$$E_{min} \leq E(n_0 - 1) + \sum_{n'=n_0}^{n} \tilde{S}(n') - \sum_{n'=n_0}^{n} g(n') \leq E_{max}, \tag{8}$$

$$g(n) \geq \max[g_{min}, \tilde{S}(n) - R_{max}], \tag{9}$$

**Algo 1:** Sensor Scheduling in epoch $n$ (SS)

```
Result: $f_k(n)$ that attains the maximum in (7), given $g(n)$.
1 Initialize: $\mathcal{K}' = \mathcal{K}$, done = FALSE;
2 if $g(n) \le g_{min} = \sum_{k\in\mathcal{K}}(E_k^0 + E_k^1 f_{k,min})$ then
      $f_k(n) = f_{k,min}$; done = TRUE;
3 while !done do
4     $E' = \sum_{k\in\mathcal{K}\setminus\mathcal{K}'}(E_k^0 + E_k^1 f_{k,min}) + \sum_{k\in\mathcal{K}'} E_k^0$ ;
5     For $k \in \mathcal{K}'$, calculate
      $f_k(n) = \frac{1}{\alpha_k}\left[\ln\left(\frac{w_k\alpha_k}{E_k^1}\right) + \frac{g(n)-E'-\sum_{k\in\mathcal{K}'}\frac{E_k^1}{\alpha_k}\ln\left(\frac{w_k\alpha_k}{E_k^1}\right)}{\sum_{k\in\mathcal{K}'}\frac{E_k^1}{\alpha_k}}\right]$;
6     $\mathcal{K}'_+ = \{k \in \mathcal{K}' : f_k(n) > f_{k,min}\}$;
7     if $\mathcal{K}'_+ = \mathcal{K}'$ then done = TRUE; else $\mathcal{K}' = \mathcal{K}'_+$;
8 end
```

*where* $n \in [n_0, N]$, $g_{min} = \sum_{k\in\mathcal{K}}(E_k^0 + E_k^1 f_{k,min})$ *and* $\tilde{S}(n) = S(n) - E_{base}$.

If $\tilde{G}$ represents the constraint set implied by Eqs. (8)-(9) then Lemma L.1 implies that regarding optimality, constraints $g \in G$ and $g \in \tilde{G}$ are equivalent when $R(n) = R_{max}$ $\forall n$.

The second observation is that due to the above approximation, the solution of Eq. (6) is independent of the exact nature of the function $V$, as long as it is an increasing, strictly concave function with $V(0) = 0$. This observation enables us to solve the EA problem without solving the SS problem first.

Further, the solution to the EA problem is a *max-min fair* solution under additional constraints [15]. More precisely, if *OPT* represents the optimal allocation vector under $g \in \tilde{G}$, then it is equivalent to a *Constrained Max-min Fair (CMF)* solution that maximizes the minimum energy allocated to the different epochs $n$ (lexicographically), subject to constraints $g \in \tilde{G}$. A more formal definition of CMF and proof of its equivalence to OPT are provided in [14].

The equivalence of OPT and CMF allows us to find OPT by developing an efficient bottleneck-based algorithm that computes the max-min fair rates using a recursive dynamic programming-like approach, exploiting the nested nature of the constraints in Eq. (8). The intuition behind our algorithm (Algorithm 2) which computes a CMF (and equivalently, an OPT) solution, is as follows. In any epoch, the algorithm pivots at the maximum battery state $\tilde{E}(n)$ (see line 1 of Algorithm 2) to avoid violating the upper bound constraint on the battery energy level ($E_{max}$). Subject to constraints in Eq. (9) the algorithm gradually increases the $g(n)$ values while satisfying the lower bound on the battery energy ($E_{min}$) at all times.

Theorem T.1. *Assuming that constraints (8)-(9) are feasible, the energy allocation vector $g$ computed by Algorithm 2 is CMF, and therefore OPT.*

In Algorithm 2 the step that dominates the complexity calculation is finding $g_m$ and $g_n$. The value of $g_m$ that satisfies $E(n_0 + 1) + \sum_{n'=n_0}^{m} \tilde{S}(n') - \sum_{n'=n_0}^{n_b} g(n') - \sum_{n'=n_b+1}^{m} \max(g(n'), g_m) = E_{min}$ can be found in $O(N)$ time. In the worst case, we run this step up to $N^2$ times; therefore the EA algorithm takes $O(N^3)$ time. Therefore the total time complexity of SEMA-A is $O(N^3 + NK^2)$.

**Algo 2:** Energy Allocation between epochs (EA)

```
Result: $g(n)$ that maximizes $\sum_{n=n_0}^{N} V(g(n))$, s. t. (8)-(9).
1  Define: $\tilde{E}(n) = E_{max}$ for $n < N$, and $\tilde{E}(N) = E_{min}$;
2  Initialize: $n_b = n_0 - 1$, $g(n) = \max[g_{min}, \tilde{S}(n) - R_{max}]$ $\forall n$;
3  for $n \in [n_0, N]$ do
4      while $E(n_0 - 1) + \sum_{n'=n_0}^{N} \tilde{S}(n') - \tilde{E}(n) > \sum_{n'=n_0}^{N} g(n')$
         do
5          Find $g_n$ satisfying: $E(n_0 - 1) + \sum_{n'=n_0}^{n} \tilde{S}(n') -$
             $\sum_{n'=n_0}^{n_b} g(n') - \sum_{n'=n_b+1}^{n} \max(g(n'), g_n) = \tilde{E}(n)$;
6          if no solution for $g_n$ then $g_n = g(n)$;
7          for $m \in [n_b + 1, n - 1]$ do
8              Find $g_m$ satisfying:
                $E(n_0 - 1) + \sum_{n'=n_0}^{m} \tilde{S}(n') - \sum_{n'=n_0}^{n_b} g(n') -$
                $\sum_{n'=n_b+1}^{m} \max(g(n'), g_m) = E_{min}$;
9              if no solution for $g_m$ then $g_m = g(m)$;
10         end
11         $g = \min(g_n, \min_{m\in[n_b+1,n-1]} g_m)$;
12         $g(n) = \max(g(n), g)$;
13         for $m \in [n_b + 1, n - 1]$ do
14             if $E(n_0 - 1) + \sum_{n'=n_0}^{m} \tilde{S}(n') - \sum_{n'=n_0}^{n_b} g(n') -$
                $\sum_{n'=n_b+1}^{m} \max(g(n'), g) \le E_{min}$ then ;
15             $n_b = m$;
16             $g(m) = \max(g(m), g)$;
17         end
18     end
19 end
```

## 3.5 Complexity of SEMA and SEMA-A

The MPC optimization approach used in SEMA is a convex optimization problem. The time complexity of convex programming solutions depends on the degree of approximation desired, e.g., gradient-based algorithms typically require $O(1/\epsilon)$ time to attain a degree of sub-optimality $\epsilon$ (which we typically want to be small). In addition, solving convex optimization problems quickly requires a solver (such as Gurobi) with a large memory/storage footprint, which may be difficult to accommodate on IoT devices. In contrast, SEMA-A is able to solve the problem to a very close degree of optimality in cubic time, as described above.

A dynamic programming-based approach (such as in [4]) could also be developed to solve the MPC problem posed in Eqs. (1)-(5). However, due to the continuous nature of the state-action space, such an approach will suffer from the curse of dimensionality. The nature of the constraints present in the problem also makes the application of dynamic programming in this case less straightforward. In fact, the recursive algorithm for Energy Allocation described in Algorithm 2 can be viewed as a dynamic programming approach, where the SEMA-A approximations and specific structure of the problem are utilized to avoid the dimensionality issue, and solve the constrained dynamic programming problem at low complexity.

Another possible approach would be to use the neural network (NN) based solutions such as [16]. The time complexity of such solutions depends on different factors such as the type of algorithm used, learning rate, number of layers and neurons per layer, etc., and is not directly comparable with SEMA or SEMA-A in terms of complexity. However, the use of NN or other machine learning (ML) based approaches is justifiable when the problem is difficult to model or its parameters are difficult to estimate. But, that is not the case here. Further, ML-based algorithms would require extensive training over different solar energy profiles, making them difficult to quickly deploy or implement in SEMA devices. The SEMA-A algorithm with lower complexity and implementation requirements is an efficient and practical solution for managing energy at the IoT device when compared to the alternative approaches.

## 4 MEASUREMENT-BASED CHARACTERIZATION

Before we can evaluate SEMA, a few supporting algorithmic components are required which we summarize at a high level here.

### 4.1 Battery Energy Management

To estimate the current battery energy level we measure the current battery voltage $v_n$ and then use voltage translation [2] and coulomb counting [18] to estimate the battery energy level. The current battery energy is given by $E(n) = \frac{E_{max}(v_n - v_{min})}{(v_{max} - v_{min})}$, where $v_{max}$ and $v_{min}$ are the maximum and minimum voltage the battery should hold. These are set to 4.2V and 3.0V respectively. Every epoch, the current battery energy level, and the solar prediction results estimate how much energy is available for the device during the rest of the day. SEMA and SEMA-A then use this information to determine the task values and predict the battery energy levels over the remaining epochs.

### 4.2 Solar Energy Prediction

We use a simple solar energy prediction approach that assumes that the daily solar energy pattern follows a parabolic curve that can be represented by $S = c - a(n - b)^2$, where $S$ is the solar energy at epoch $n$, the peak energy $c$ occurs at epoch $n = b$ (mid-day), and $a$ is an appropriately chosen constant. At every epoch, we run the solar prediction algorithm to determine new values for $a$ and $c$ and thus generate a new parabolic curve whose peak can be higher or lower than the previous epoch's curve. From this curve, we can estimate the total solar available for the rest of the day. Fig. 2 illustrates the concept and shows the predicted solar curves $S$ and $S'$ at epoch $n - \delta$ and $n$ respectively. $\delta$ is the number of epochs over which we use the history of the battery energy to estimate the amount of solar energy available during this epoch. The new values for $a$ and $c$ at epoch $n$ are $a'$ and $c'$. These are calculated using Eq. 10 where, $E'(n - \delta)$ and $E'(n)$ are the measured battery energy at epochs $n - \delta$ and $n$. The battery energy values are calculated by measuring the battery voltage (see Section 4.1). Also, $\delta = 1$, since we run the solar prediction every epoch and $F(n')$ denotes the total amount of energy used by the SEMA Stick in epoch $n' \in [n - \delta, n - 1]$ to run all the tasks and provide base power for the device.

$$c' = a'b^2; \quad a' = \frac{E'(n) - E'(n-\delta) + \sum_{n-\delta}^{n-1} F(n')}{\delta b^2 - \left(\frac{(n-b)^3}{3} - \frac{(n-\delta-b)^3}{3}\right)} \tag{10}$$

Further details are provided in [14]. We only use the prediction values over short time periods since the solar prediction is updated every epoch. This is still effective because the frequent adaptation captures any energy changes and adjusts the prediction accordingly. Our solar prediction can give the same result for all IoT devices in an area, but also gives unique customized solar energy predictions based on the solar energy received at each IoT device and its battery state changes (e.g., account for obstructions that impact a particular device's solar energy, even if it is for one or a few epochs).

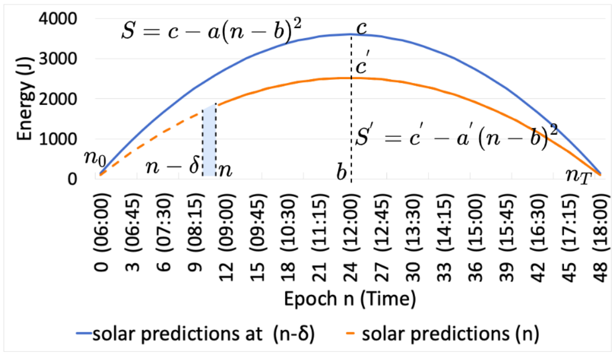

**Figure 2: Solar energy prediction (at epochs $n - \delta$ and $n$).**

### 4.3 Task Utility and Energy Models

To measure the information utility for each task $k$ we use a simple concave utility function of the form $U_k = 1 - e^{-\alpha_k f_k}$. Here, $U_k$ is the calculated utility value for the task (value between 0 and 1), $f_k$ is the variable parameter of the task, and $\alpha_k$ is a utility constant that is task-dependent and determined through tuning. This simple utility function increases as the task parameter value increases and the utility flattens out beyond a certain value of the task key variable, depending on the task. The value of $\alpha_k$ is selected so the utility even for the lowest value of $f_k$ yields a practically useful sensor input. Thus, the information utility at the lowest task variable value for all the tasks is ~60% of the maximum information utility. For example, for the video task where the task variable parameter is video duration, longer videos have higher information utility, the utility flattens out as we approach the maximum video duration (i.e. 30s) and at the lowest video duration (5s) the utility is 0.632.

By adjusting the task variable parameters and measuring their energy use experimentally, the energy requirements for each task were determined and modeled. SEMA and SEMA-A algorithms described in Sections 3.4.1 and 3.4.2 use these energy and utility models when determining the appropriate task parameters. Please refer to [14] for more details regarding the models.

## 5 RESULTS AND EVALUATION

### 5.1 Metrics

The performance metrics we used capture how long the device can run, and the task values used. Since our energy goal is to ensure the device battery energy is at least at the minimum level or higher by 06:00 the next morning, we monitor and compare the actual measured battery ($E_a$) and the predicted battery level ($E_p$). For the task values we compare task values selected with SEMA and SEMA-A when using the predicted battery and actual battery energy levels.

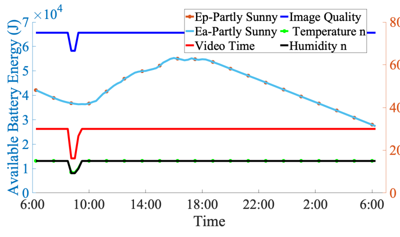

Figure 3: SEMA performance [Day 1 (Partly sunny)].

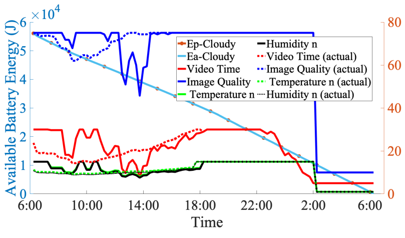

Figure 4: SEMA performance [Day 2 (Cloudy)].

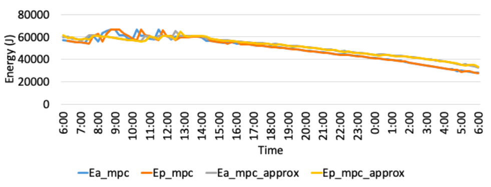

Figure 5: SEMA (MPC) and SEMA-A (MPC Approx) Energy [Day 3 (Sunny)].

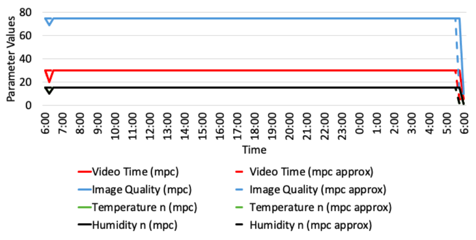

Figure 6: SEMA (MPC) and SEMA-A (MPC Approx) Tasks [Day 3 (Sunny)].

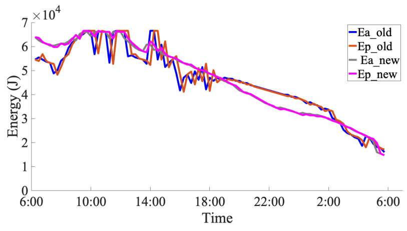

Figure 7: Comparing performance with old and new batteries with SEMA (Energy).

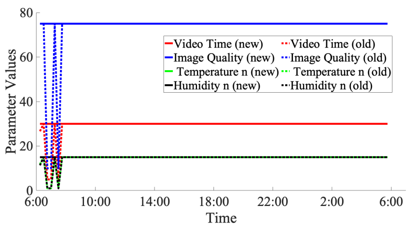

Figure 8: Performance with old vs. new batteries with SEMA (Task parameters).

## 5.2 Experimental Results

We placed a few SEMA sticks in an outdoor environment and obtained solar data and battery measurements for different scenarios i.e., sunny day, cloudy day, etc. Using these measurements and solar profiles, we run the SEMA algorithm to determine what the performance of SEMA would be on such days. We present the results for two days; a Partly Sunny day (Day 1 (Partly Sunny), and a Cloudy day (Day 2 (Cloudy)).

Fig. 3 shows the results for the Partly Sunny day (Day 1) where the solar energy was sufficient to charge the battery significantly. SEMA selects maximum task values most of the time. However, there is some task adaptation required in the morning around 09:00, when energy for the SEMA Stick is still derived from the battery, due to low solar energy. The resulting low solar energy prediction causes SEMA to select lower task values. However, after two epochs, when the battery starts charging, higher solar energy predictions are made, and SEMA adapts to use higher task values. This allows the battery to last throughout the night.

Fig. 4 shows the results when using the Cloudy day solar profile (Day 2). Both the measured battery level $E_a$ and predicted battery level $E_p$ decrease during the day since the solar energy is insufficient to charge the battery. The graph also shows the task values selected based on $E_p$ (solid lines) and $E_a$ (dashed lines). The task curves generally follow the same pattern indicating that the SEMA predicted task values are close to what is possible with the actual measured energy. In cases where the predicted task values are higher than what is actually possible (see video task from 06:00-12:00), SEMA auto corrects for this soon thereafter in the afternoon (see 12:00-18:00) by reducing the task values to save energy. This shows that by adapting every epoch, SEMA is able to adjust to changes in the battery state and solar energy availability to ensure that the device's energy goals are met.

## 5.3 Comparing SEMA-A with SEMA

Two SEMA Sticks were placed in an outdoor environment over a period of several months with one SEMA Stick running SEMA and the second running SEMA-A. We show results for one sunny day (Day 3 (Sunny)) where the solar energy during the day was high.

Fig. 5 compares $E_a$ and $E_p$ with SEMA (which runs the exact MPC) and SEMA-A (which runs the MPC approximation). In both cases $E_a$ and $E_p$ values are fairly close. The variations in the battery levels in the morning are due to changes in the solar energy available for charging the battery. The initial battery energy level was high and the solar energy during the day was also high enough to supply most of the energy needed for the sensors and for idle power. Therefore, in both cases, the battery was able to last throughout the night till the next morning. Fig. 6 shows the task values selected. Due to sufficient energy availability (high battery energy and high solar), maximum task values were used most of the time, with two notable exceptions. At around 06:15, SEMA selected lower task values. This is because the Stick running SEMA had a slightly lower battery level (unlike the stick running SEMA-A), so it adapted the task values accordingly for that epoch. The second exception was at 06:00 the next morning. As a new 24-hour period begins, both algorithms try to determine the task values to ensure the battery lasts for the next 24 hours. Since there was no solar overnight, the predicted solar is low and the current battery is not full. Thus, both algorithms switch to use minimum task parameters for this epoch. Both algorithms also have a similar battery level prediction error rate of approximately 1.66% and 1.14% for SEMA and SEMA-A respectively.

## 5.4 Adaptability of SEMA with Aging Batteries

As batteries age, their performance sometimes degrades in an unpredictable manner. Since SEMA adapts to the current solar energy availability and battery state we experimentally compare the performance of SEMA with new and old batteries. The 3-year-old batteries were used regularly to run SEMA experiments including most of the experiments in this paper while the new batteries had seen limited use. We placed two SEMA sticks (with old and new batteries) next to each other on a sunny day and ran the MPC algorithm over a 24-hour period.

From Fig. 7 the predicted and measured battery values with the new batteries reduce much more smoothly. Due to age, the older batteries experience more voltage spikes which affect the battery state prediction. In addition, since the old battery can no longer charge fully, once the device is removed from a non-renewable power source at the beginning of the first epoch, the voltage drops significantly (compare the old and new results at 06:00 where the new battery maintains the charge). The task values selected are compared in Fig. 8. Since this was a sunny day, both SEMA Sticks were able to charge their batteries and could run with maximum task values for the full 24-hour period, although the SEMA Stick with the old battery had to adapt task parameters at the beginning of the day due to the voltage drop mentioned earlier. These results show that even when an IoT device experiences unexpected energy variations due to the hardware, e.g., due to battery aging, SEMA adapts to these changes and selects task values that enable the device to achieve its energy goals.

## 6 CONCLUSIONS

Multi-sensor IoT devices enable monitoring of different phenomena over large areas. Such devices typically use renewable energy sources and batteries, and need energy-efficient management solutions that maximize both device lifetime and information utility. This work presents SEMA, an adaptive and efficient energy management solution for IoT. SEMA uses the current battery state, predicted available solar energy, task energy, and utility models with an MPC-based approach to dynamically determine appropriate task values that maximize information utility while ensuring device energy goals are met.

Since the MPC approach has to run on a centralized sink and download task values to the IoT device, it is compute and communication intensive. We propose SEMA-A, an efficient MPC approximation approach that is simple enough to be run on the IoT device itself. SEMA-A decomposes the original MPC optimization problem into an energy allocation problem (splitting available energy into all epochs) and a sensor scheduling problem (dividing the epoch energy between all available tasks). Experimental results show that SEMA adapts the task values based on the available energy, and that SEMA and SEMA-A have very similar performance.

## ACKNOWLEDGMENTS

This work was supported by the US National Science Foundation grant CNS-1818971 and the US Dept. of Commerce, NIST PSIAP award 70NANB17H188.

## REFERENCES

[1] Joshua Adkins, Branden Ghena, Neal Jackson, Pat Pannuto, Samuel Rohrer, Bradford Campbell, and Prabal Dutta. 2018. The signpost platform for city-scale sensing. In *2018 17th ACM/IEEE International Conference on Information Processing in Sensor Networks (IPSN)*. IEEE, IEEE, Porto,Portugal, 188–199.

[2] Davide Andrea. 2009. *White Paper - Estimating the State Of Charge of Li-Ion batteries*. Elithion. Retrieved June 2, 2021 from http://liionbms.com/php/wp_soc_estimate.php

[3] Raffaele Carli, Graziana Cavone, Sarah Ben Othman, and Mariagrazia Dotoli. 2020. Iot based architecture for model predictive control of hvac systems in smart buildings. *Sensors* 20, 3 (2020), 781.

[4] A. Caruso, S. Chessa, S. Escolar, X. del Toro, and J. C. López. 2018. A Dynamic Programming Algorithm for High-Level Task Scheduling in Energy Harvesting IoT. *IEEE Internet of Things Journal* 5, 3 (2018), 2234–2248. https://doi.org/10.1109/JIOT.2018.2828943

[5] Danco Davcev, Kosta Mitreski, Stefan Trajkovic, Viktor Nikolovski, and Nikola Koteli. 2018. IoT agriculture system based on LoRaWAN. In *2018 14th IEEE International Workshop on Factory Communication Systems (WFCS)*. IEEE, Imperia, Italy, 1–4. https://doi.org/10.1109/WFCS.2018.8402368

[6] UC San Diego. 20122. *High Performance Wireless Research and Education Network (HPWREN*. UC San Diego. Retrieved Jul 21, 2022 from http://hpwren.ucsd.edu/cameras

[7] Carlos E. Garcia, David M. Prett, and Manfred Morari. 1989. Model predictive control: Theory and practice - A survey. *Automatica* 25, 3 (1989), 335 – 348. https://doi.org/10.1016/0005-1098(89)90002-2

[8] Lei Hang and Do-Hyeun Kim. 2018. Enhanced model-based predictive control system based on fuzzy logic for maintaining thermal comfort in IoT smart space. *Applied Sciences* 8, 7 (2018), 1031. https://www.mdpi.com/2076-3417/8/7/1031

[9] Robert Hartung, Ulf Kulau, Björn Gernert, Stephan Rottmann, and Lars Wolf. 2017. On the experiences with testbeds and applications in precision farming. In *Proceedings of the first ACM international workshop on the engineering of reliable, robust, and secure embedded wireless sensing systems*. 54–61.

[10] Texas Instruments. 2015. *Characteristics of Rechargeable Batteries*. Texas Instruments. Retrieved Jan 29, 2022 from https://www.ti.com/lit/an/snva533/snva533.pdf

[11] Q. Ju and Y. Zhang. 2018. Predictive Power Management for Internet of Battery-Less Things. *IEEE Transactions on Power Electronics* 33, 1 (2018), 299–312. https://doi.org/10.1109/TPEL.2017.2664098

[12] Aman Kansal, Jason Hsu, Mani Srivastava, and Vijay Raghunathan. 2006. Harvesting Aware Power Management for Sensor Networks. In *Proceedings of the 43rd Annual Design Automation Conference* (San Francisco, CA, USA) *(DAC '06)*. Association for Computing Machinery, New York, NY, USA, 651–656. https://doi.org/10.1145/1146909.1147075

[13] Mohamed Rawidean Mohd Kassim, Ibrahim Mat, and Ismail Mat Yusoff. 2019. Applications of Internet of Things in Mushroom Farm Management. In *2019 13th International Conference on Sensing Technology (ICST)*. 1–6. https://doi.org/10.1109/ICST46873.2019.9047702

[14] E. Liri, K. K. Ramakrishnan, K. Kar, G. Lyon, and P. Sharma. 2022. *Appendix: An Efficient Energy Management Solution for Renewable Energy Based IoT Devices*. https://tinyurl.com/2p942x6m

[15] Hanan Luss. 1999. On Equitable Resource Allocation Problems: A Lexicographic Minimax Approach. *Operations Research* 47, 3 (1999), 361–378. https://doi.org/10.1287/opre.47.3.361

[16] Ambarish G. Mohapatra and Saroj Kumar Lenka. 2016. Neural Network Pattern Classification and Weather Dependent Fuzzy Logic Model for Irrigation Control in WSN Based Precision Agriculture. *Procedia Computer Science* 78 (2016), 499–506. https://doi.org/10.1016/j.procs.2016.02.094

[17] Honglin Ren and Kwan-Wu Chin. 2021. Novel Tasks Assignment Methods for Wireless Powered IoT Networks. *IEEE Internet of Things Journal* 9, 13 (2021), 10563–10575. https://doi.org/10.1109/JIOT.2021.3121415

[18] Nareg Sinenian and Daniel Shai. 2017. 3 - Advances in Power Converters. In *The Power Grid*, Brian W. D'Andrade (Ed.). Academic Press, San Diego, California, 57–92. https://doi.org/10.1016/B978-0-12-805321-8.00003-3

[19] Antonio Valente, Sérgio Silva, Diogo Duarte, Filipe Cabral Pinto, and Salviano Soares. 2020. Low-Cost LoRaWAN Node for Agro-Intelligence IoT. *Electronics* 9, 6 (2020). https://doi.org/10.3390/electronics9060987

[20] Deepak Vasisht, Zerina Kapetanovic, Jongho Won, Xinxin Jin, Ranveer Chandra, Sudipta Sinha, Ashish Kapoor, Madhusudhan Sudarshan, and Sean Stratman. 2017. FarmBeats: An IoT Platform for Data-Driven Agriculture. In *14th USENIX Symposium on Networked Systems Design and Implementation (NSDI 17)*. USENIX Association, Boston, MA, 515–529.

[21] Alert Wildfire. 2022. *Alert Wildfire*. University of Nevada. Retrieved Jul 21, 2022 from http://www.alertwildfire.org