Adaptive Power Management in Solar Energy Harvesting Sensor Node using Reinforcement Learning

SHASWOT SHRESTHAMALI, The University of Tokyo
MASAAKI KONDO, The University of Tokyo
HIROSHI NAKAMURA, The University of Tokyo

In this paper, we present an adaptive power manager for solar energy harvesting sensor nodes. We use a simplified model consisting of a solar panel, an ideal battery and a general sensor node with variable duty cycle. Our power manager uses Reinforcement Learning (RL), specifically SARSA(λ) learning, to train itself from historical data. Once trained, we show that our power manager is capable of adapting to changes in weather, climate, device parameters and battery degradation while ensuring near-optimal performance without depleting or overcharging its battery. Our approach uses a simple but novel general reward function and leverages the use of weather forecast data to enhance performance. We show that our method achieves near perfect energy neutral operation (ENO) with less than 6% root mean square deviation from ENO as compared to more than 23% deviation that occur when using other approaches.

CCS Concepts: • **Computing methodologies → Machine learning approaches**; • **Hardware → Energy generation and storage**; **Power estimation and optimization**; • **Computer systems organization →** *Embedded and cyber-physical systems*;

Additional Key Words and Phrases: Wireless sensor nodes, reinforcement learning, power management, IoT

ACM Reference format:
Shaswot Shresthamali, Masaaki Kondo, and Hiroshi Nakamura. 2017. Adaptive Power Management in Solar Energy Harvesting Sensor Node using Reinforcement Learning. *ACM Trans. Embedd. Comput. Syst.* 17, 4, Article 39 (October 2017), 22 pages.
DOI: 0000001.0000001

1 INTRODUCTION

Energy harvesting sensor nodes (EHSN) are sensor nodes equipped with an energy buffer (battery) and an energy harvesting module. The presence of a battery along with an energy harvesting module theoretically allows for perpetual operation limited only by the lifetime of the hardware [8]. Perpetual operation is critical for realizing pervasive computing and Internet of Things (IoT) as it opens the possibility of autonomous operation of sensor nodes.

Before the integration of energy harvesting modules,optimization techniques revolved around minimizing power consumption of the node to extend the lifetime of the battery [20] and [22].

This work was partially supported by JSPS KAKENHI Grant Number 16K12405. The first author acknowledges the Japanese Government (MEXT) Scholarship received for his study in The University of Tokyo.
Authors' address: Nakamura Laboratory, Room 508, 5F, Engineering Building 1, The University of Tokyo, Hongo 7-3-1, Bunkyo, Tokyo, JAPAN, 113-8656
This article was presented in the International Conference on Embedded Software 2017 and appears as part of the ESWEEK-TECS special issue.
 1539-9087/2017/10-ART39 $15.00
DOI: 0000001.0000001

However, with the possibility of harvesting energy from the ambient environment, the node is able to recharge its battery. During times when energy cannot be harvested, the node can operate on the energy stored in the battery.

This leads to the concept of energy neutral operation (ENO), introduced in [10], [16] and [9]. ENO is achieved when the energy consumed by the node is less than or equal to the energy harvested by the environment. In addition, we would also like to fully utilize all of the energy harvested to power the sensor node. The condition when the amount of energy harvested equals the amount of energy consumed by the node is termed as *node level energy neutrality*[19] or ENO-Max condition [23]. Achieving node level energy neutrality ensures a minimum level of operation by the sensor node at all times. In addition to this, the node is also able to increase its utility by exploiting as much of the harvested energy when possible [9].

Achieving node level energy neutrality comes with several constraints, namely the size of the battery and the maximum/minimum rates at which energy can be consumed and harvested. In addition, the energy harvested from the environment is often unpredictable and unreliable. In such a context, achieving node level energy neutrality is not trivial.

A major challenge in achieving node level energy neutrality is using the correct power management strategy to accommodate for the changing energy harvesting profiles. Sensor nodes such as those strapped to animals or mobile sensor nodes will encounter diverse and varying energy harvesting opportunities. In addition, changes in weather patterns and climate will also require the sensor nodes to adopt a suitable power management strategy. The sensor node must also be able to adjust its behavior to account for changes in its device parameters such as changes in energy harvesting efficiency, battery degradation [14] and partial node failure/decrease in node's energy efficiency. It is not practical to prepare heuristic contingency plans for all situations. This problem is even more aggravated when we have to deal with billions and trillions of sensor nodes, each with a unique power consumption profile and deployed to a unique environment. A natural solution in this case is to have sensor nodes that are capable of autonomously learning optimal strategies and adapting once deployed in the environment.

Several adaptive approaches to power management have been previously proposed. Section 2 goes into more details about these methods. Our paper is based on these approaches and overcomes some of their limitations. The first formal solution was presented in [9]. The authors use optimization techniques based on linear programming integrated with an energy prediction mechanism. In [23], the authors propose a linear quadratic control system for adaptive power management. Reinforcement learning (RL) for power management is used in [5],[6], [2] and [15]. More details on RL are given in Section 3. In short, in a RL setting, the power manager(or the agent) executes some actions and interacts with its environment. The environment responds by awarding the agent with a single value scalar reward according to some reward function. Through a number of such interactions, the agent searches its action space and memorizes (learns) the most favorable action for a particular state the agent might be in. An RL based learning approach is adaptive by nature of its learning [13]. Since its learning is integrated with interaction with its environment, the agent is able to respond to changes in slowly varying non-stationary environments [21].

We propose a power management policy based on tabular SARSA(λ) reinforcement learning (RL) [21] for a solar EHSN. In this method of RL, some fraction of the reward is back-propagated to all the actions and states that contributed to its achievement. This back propagation of feedback results in faster learning [21] as compared to other methods that propagate their rewards by only one step. In our formulation of the problem, the node is able to dynamically adapt its power consumption depending on the energy harvesting opportunities by varying its duty cycle to ensure ENO. The power manager makes its decisions based on information about its *distance from energy neutrality,*

battery level, amount of energy being harvested and *the weather forecast for the day*. The learning theory and its implementation methodology are explained in more detail in Section 4 and Section 5. The distance from energy neutrality is referred here as Energy Neutral Performance (ENP). One of our main contributions is the inclusion of ENP in the state definitions. This dramatically reduce the learning time and enables the agent to be highly adaptive to environmental changes. Our results, described in Section 6 show that the agent is able to adapt to seasonal variations, changes in climate due to change in location, battery degradation and changes in device working parameters . As to our current knowledge, the general adaptivity of a RL based power management strategy for EHSN has not been investigated before.

The reward function is extremely critical in a RL framework as it is responsible for educating the agent on what kind of behaviors are favorable. [2] and [15] use metrics related to data transmission as rewards. This makes sense when the power consumption of data transmission is significantly greater than that of sensing operations. This assumption limits the scope of application and excludes situations where other critical tasks consume significant power. For example, when a mobile sensor node has to allocate power consumption between locomotion, processing, transmission and sensing, the amount of data transmitted may not be a good basis to determine the optimality of the actions. The reward function mentioned in [6] has a more general scope of application. Here, the author propose a reward function based on the instantaneous battery level and ENP. In our paper, we improve on this and introduce a simplified reward function that is natural, intuitive and performs satisfactorily. We use the ENP at the *end of an episode(day)* as the sole basis of our reward function. This is a novel contribution of our work. An added benefit to this is that this reduces the variance in battery levels and this in turn contributes to increase in battery life [1]. The novelty of this reward definition is that it is indifferent to the node's tasks that consume power. As a result, our proposed power management policy has a wider scope of application. In this paper, we assume all the power consumed by the node is for sensing purposes. This is only for the sake of example and we can extend this to any kind of node operation.

Our final contribution is to take into account the information about the weather forecast to enhance performance. [18] use forecast data to model a weather predictor. It is obvious that the availability of a perfect weather oracle would greatly enhance the performance of a power manager. [9] and [2] mention some power management strategies when such non-causal data is available. We use the formulation in [9] to obtain the theoretical upper limit in performance and use it for comparison. It is impossible to expect such non-causal information in practice. However, it is possible to obtain some general indication of future weather for the day. The information about the type of weather - for e.g. sunny, fair, overcast etc. can be easily acquired from weather websites and apps. We make use of this information to increase the performance of our agent. For the system presented in this paper, we use the forecast data to give an estimate of expected solar energy. However, the same argument can be made for other EHSN such as those driven by wind power.

In summary this work presents the following contributions:

- **Adaptivity:** We propose a SARSA(λ) RL based power management approach to ensure ENO-Max operation. This approach enables the power manager to adapt to changes in weather, climate, battery degradation and device parameters while achieving faster convergence. This is attributed to our novel idea of using ENP as a state definition parameter.
- **General Scope of Application:** A direct consequence of our unique reward function is that the power manager policy can learn to accommodate any kind of realistic system. As a result, the scope of application is more general. Using our method, the policy can adapt accordingly and maintain near-optimal performance even if the system parameters are to change over time.

- **Enhanced performance:** Our power management strategy improves performance by as much as four times by leveraging information about the general weather forecast. The use of such data allows the agent to anticipate the amount of energy that can be harvested and adopt a suitable power management strategy.

2 RELATED WORK

An overview of different energy harvesting architectures, sources of ambient energy and real world example of EHSN is presented in [19]. In [4], the authors elaborate on design challenges and solutions for energy harvesting communication systems in predictable and unpredictable environments. [11] surveys management strategies for wireless sensor nodes on basis of their energy provision and consumption.

The first formal description of energy harvesting sensor nodes and ENO was done in [9]. The authors in [9] present a linear programming optimization solution when non-causal information about the environment is available. We adapt their optimization technique to determine the optimal policy and compare it with our results. When non-causal information is not available, they predict the amount of energy that is expected and decide on the duty cycle accordingly. They take into account battery inefficiencies and also allow for adaptation of the duty cycle to accommodate for any changes in predicted and actual harvesting conditions. However this technique relies heavily on the accuracy of its prediction mechanism and prior knowledge of the statistics of the environment. This is not always a realistic assumption.

In [23], the authors approach this problem by specifying a battery-centric objective function. They argue that by minimizing the average square deviation in battery level, ENO-Max condition is satisfied. They achieve this by using a linear quadratic-tracker . They also take into account the minimization of duty cycle variance. They do not attempt to model the energy source and do not require prior information about the environment. While this approach is adaptive in nature, it requires careful calibration of hyper parameters depending upon the environment. The agent would not be able to "fix" its own hyper parameters if its environment should change.

An adaptive duty cycling method using continuous time Markov Chain Modeling by taking into account the Quality of Service (QoS) requirements and the rate of change of battery level depending upon the battery chemistry is given in [3]. RL strategies for power management are described in [17]. In [2], the authors tackle the case of point to point wireless communication system with stochastic data arrival and channel state processes. They provide optimization techniques when non-causal information about the environment is known and in cases when statistics of the environment are known beforehand. They then propose an RL based learning theoretic approach to maximize total transmitted data. The authors use discrete states, actions and rewards similar to our approach. In [15], the authors improve on this by using function approximation with RL to accommodate for continuous states and rewards. As mentioned before, the techniques proposed in [2] and[15] use data transmission metrics as a basis for rewards. This is not always reasonable in general applications. Fuzzy rewarding schemes are mentioned in [12] and [7].

Optimizing the duty cycle to achieve node neutrality using RL is proposed also in [5], [6]. Their basis for reward is the distance from energy neutrality and the current battery level. While this is a more general formulation of the reward function, it is quite handcrafted. The reward function is an indication of what kind of states or actions are favorable rather than directives on *how* to achieve a certain goal. The distance from energy neutrality is not a good measure of reward *during* an episode. We also use the same metric to determine the reward. However, the reward is fed back only at the *end* of the episode. We believe that this reflects the ENO-Max objective better.

None of the previous research have investigated the adaptivity of their algorithms to changes in climatic conditions and battery degradation. We observe the behavior of our agent to various environments and observe its adaptive performance.

[18] show that performance of EHSN can be improved by using weather forecast information. Their work uses this information to make better predictions about the energy that can be harvested. Along the same lines, we also leverage the use of easily obtainable forecast data for a particular day to improve performance.

3 THEORETICAL BACKGROUND

In this section, we explain our system model. We then go briefly into RL and the SARSA algorithm.

3.1 System Model

Our system model contains four components - an energy harvesting source (solar panel), an energy buffer (battery), a sensor node (load) and a power management unit that controls the node power consumption as illustrated in Figure1. The battery has a maximum capacity of B_{MAX}. We assume the sensor node to have a variable duty cycle. The power management unit uses information about the current battery level, energy being harvested and the information about the weather to decide the duty cycle of the node. We further assume that a higher duty cycle implies a higher power consumption and higher performance from the sensor node.

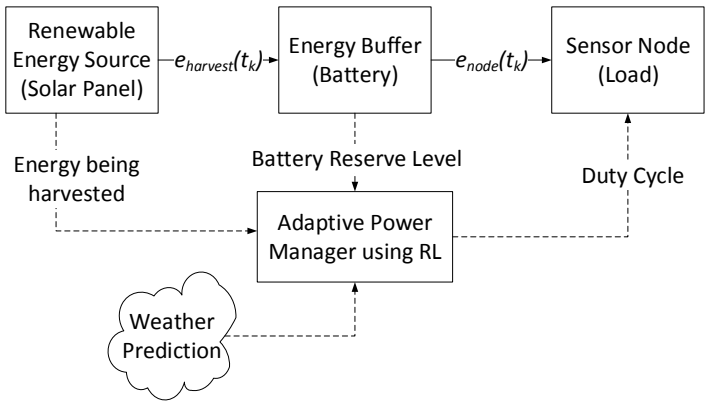

Fig. 1. System Model

We use a discrete time model where time is divided into equal intervals called epochs. A set of epochs constitutes an episode. In the beginning of each epoch, the power manager receives information about the weather condition that can be expected for the day. $e_{batt}(t_k)$ is the battery energy level at the start of the epoch t_k. During each epoch, t_k, the system receives a finite amount of energy, $e_{harvest}(t_k)$, from the environment. The power manager decides on a duty cycle, $d(t_k)$. The sensor node consumes $e_{node}(t_k)$ amount of energy depending upon $d(t_k)$. As a result, the energy in the battery at the start of the t_{k+1} epoch is given by:

$$e_{batt}(t_k + 1) = e_{batt}(t_k) + e_{harvest}(t_k) - e_{node}(t_k) \quad (1)$$

We assume no losses due to battery inefficiencies. This can be assumed without loss of generality because any such losses can be lumped together as increase in node power consumption. The

distance from energy neutrality of the sensor node, or ENP, in a particular epoch t_k, $e_{dist}(t_k)$ is given as follows:

$$e_{dist}(t_k) = e_{harvest}(t_k) - e_{node}(t_k) \tag{2}$$

In [9], the authors present mathematical guidelines to calculate the battery size and the required battery level to account for the variance in the energy harvested by the system and consumed by the load. We follow their formulation in to determine the size of the battery and the optimum initial battery level required for ENO.

3.2 Reinforcement Learning

Reinforcement learning is a machine learning technique where the machine (or agent) learns through experience rather than through instruction. The agent interacts with an environment and receives feedback in terms of a scalar reward signal. Through trial and error, the agent learns which actions are favorable depending upon which state it might find itself in. With enough experience, we expect the agent to come up with an optimal policy that will maximize its long term cumulative reward.

A general RL model consists of a finite state space S, an agent capable of executing a set of actions A, and an environment that reacts to the said actions. The environment reacts to the agent's choice of action expressed by a scalar reward signal defined by the reward function $R : S \times A \rightarrow R$. The action, a, to be executed when the agent is in a particular state, s, is dictated by the policy, $\pi = \{(s, a)|a \in A, s \in S\}$. This is denoted by $\pi(s) = a$.

Each agent-environment interaction occupies an epoch. At epoch t_k, the agent is in state $s_k \in S$. The agent takes an action, $a_k \in A$, according to some policy π. The environment reacts to this action by changing the agent's state to a new state $s_{k+1} \in S$ and rewarding the agent with some scalar reward r_k.

The objective of the agent is to find a policy that maximizes at each time step the expected discounted sum of future reward. An optimal policy, π^*, maximizes this quantity for all state-action pairs.

To give a measure of the the goodness of a particular action, a, according to some policy π, when the agent is in some state s, we assign each state-action pair a Q-value [21]. The Q-value, $Q^\pi(s, a)$ is defined as the expected sum of discounted rewards starting from state s, taking action a and following policy π thereafter. Mathematically it is expressed as,

$$Q^\pi(s, a) = E\left[\sum_{k=0}^{N-1} \gamma^k r(s_k, a_k)\right] \tag{3}$$

where $s_0 = s, a_0 = a$, $a_k = \pi(s_k)$. γ, $0 < \gamma < 1$, is a discount factor. Equation 3 assumes that the agent-environment interaction lasts for N epochs. The Q-values for the optimal policy π^* is denoted by Q^*. In this paper, we use SARSA(λ) algorithm to learn the Q-values. Determining the optimal actions is trivial once Q^* is known. For each state s, the action a that maximizes $Q^*(s, a)$ is the optimal action. Choosing actions in this way is called a *greedy policy*.

3.2.1 SARSA:. SARSA stands for State-Action-Reward-State-Action. The Q-value, $Q^\pi(s, a)$, for a state-action pair (s_k, a_k), corresponding to a policy π, is estimated by considering the agent's transition to another state-action pair (s_{k+1}, a_{k+1}) and the reward, r_k, it receives in the process.

The agent starts out in state s_k, takes action a_k according to some policy π. As a result, it receives a reward r_k and is transported to another state s_{k+1}. The agent then considers taking the next action a_{k+1} according to the policy π. At this point, $Q(s_k, a_k)$ is updated as follows:

$$Q^\pi(s_k, a_k) \leftarrow (1 - \alpha)Q^\pi(s_k, a_k) + \alpha\left[r_k + \gamma Q^\pi(s_{k+1}, a_{k+1})\right] \tag{4}$$

where α, $0 < \alpha < 1$, is the learning factor.

We maintain a Q-table of all possible state-action pairs and their corresponding Q-values. With a Q-table, finding the optimal action requires only a single lookup. This is not a computationally intensive process once the Q-values have been sufficiently learned.

3.2.2 Eligibility Trace: When an agent passes through a series of states by performing a sequence of actions and receives a reward at the end of an episode, assigning credit to the appropriate state-action pairs becomes an issue. To resolve this, we introduce a memory variable for each state-action pair called the *eligibility trace*. The eligibility trace for a state-action pair at epoch t_k is denoted by $e_k(s, a) \in \mathbb{R}_{\geq 0}$. During each epoch, the eligibility trace for *all* state-action pairs decays by $\gamma\lambda$, and the trace for the state-action pair visited on epoch t_k is incremented by 1, i.e.

$$e_k(s,a) = \begin{cases} \gamma\lambda e_{k-1}(s,a) & \text{if } (s,a) \neq (s_k, a_k) \\ \gamma\lambda e_{k-1}(s,a) + 1 & \text{if } (s,a) = (s_k, a_k) \end{cases} \tag{5}$$

for all (s, a), where λ, $0 < \lambda < 1$, is a parameter that allows us to specify the strength with which Q-values of early state-action pairs are updated as a consequence of the final reward. The value of $e(s, a)$ is a measure of how influential the state-action pair (s, a) was in obtaining the reward at the end of an episode. We combine the concept of eligibility traces with SARSA learning to get SARSA(λ). This is explained in more detail in the following section.

3.2.3 ϵ-greedy policy: We initialize the Q-table optimistically [21] by assigning all state-action pairs high Q-values so that the algorithm is encouraged to explore more during the beginning of the training period[21]. With sufficient training, the Q-values should converge to their true values. SARSA learning converges to an optimal Q table when all state-action pairs have been visited infinitely often. However, acting greedily before convergence may lead to sub-optimal policies because the agent would not have had the opportunity to sample state-actions pairs that might have led to higher returns. In order to avoid this, we follow an ϵ-greedy policy. This mean the actions are chosen greedily most of the time, but with probability ϵ, a random exploratory action is selected.

4 SARSA FOR ENO

In this section, we explain how we set up the RL framework for our problem. We also present the algorithm for SARSA(λ) learning.

4.1 RL Framework

The RL framework defines the state space and the action space. The environment, in this context, consists of a stochastic energy source and the battery. It reacts to the agent by specifying a reward based on the duty cycle chosen and a new state depending upon the amount of energy harvested, the reserve battery level and the weather forecast information.

We train the agent in episodes. The agent takes a sequence of actions and traverses through a series of states until the end of the episode (in our case an episode consists of 24 epochs). At the end of the episode, the agent is awarded some reward which it uses to evaluate its actions and upgrade its policy (Q-table). The state definitions, action space and the reward scheme are described as follows.

4.1.1 State Space. Given the statistics of the environment, the optimum battery level, B_0, can be determined by using the formulation in [9]. What this means is that if the agent starts out with B_0 amount of battery at the start of every day, it is able to accommodate for the days with the least energy harvested as well as days with the maximum energy harvested without depleting

the battery completely or allowing it to be overcharged. We use this battery level to calculate the distance from energy neutrality. Ideally, we would like the agent to begin everyday with B_0 amount of battery and end with exactly B_0 remaining. If the battery level at epoch t_k is $B(t_k)$, we calculate the distance from energy neutrality or ENP, $e_{dist}(t_k)$, using B_0 as reference according to Equation 6.

$$e_{dist}(t_k) = B(t_k) - B_0 \tag{6}$$

$e_{dist}(t_k)$ is used to determine $S_{dist}(t_k)$, the state of distance from energy neutrality. Using this information for state definition makes our approach largely independent of the actual battery capacity and thus more generally applicable. However, the agent also takes into account whether the battery is in danger of being overcharged or completely depleted. This is reflected in $S_{batt}(t_k)$, the state of the battery reserve level. State $S_{day}(t_k)$ gives an indication of what kind of weather the agent may expect. With this state information, the agent is able to use different strategies to achieve ENO depending on how much energy it can expect to harvest. Finally, the state of the amount of energy being harvested during the present epoch is given by $S_{eharvest}(t_k)$. Hence the different states in which the agent can exist is given by combination of $S_{batt}(t_k)$, $S_{dist}(t_k)$, $S_{eharvest}(t_k)$, and $S_{day}(t_k)$ i.e.

$$(S_{batt}(t_k), S_{dist}(t_k), S_{eharvest}(t_k), S_{day}(t_k)) \in S$$

4.1.2 Action Space. The action space, A, is defined as the set of discrete duty cycles that can be chosen. $A \in (D_{min}, D_{max})$, where D_{min} and D_{max} are the minimum and maximum duty cycle of the sensor node. The agent chooses one duty cycle in each epoch.

4.1.3 Reward Function. The reward indicates what kind of behavior best serves our objective. In our RL model, the reward awarded at the end of an episode depends simply on the distance from energy neutrality at the end of the episode. The "goodness" of any action in the middle of an episode cannot be judged without taking the effect of other actions in the episode. Hence we wait until the end of the episode to judge the "goodness" of the sequence of actions chosen. This is also a fair reward system considering that ENO-Max operation can be achieved by a number of different methods i.e. it is not always necessarily the case that there is only one unique optimal policy. This is because we have not included battery inefficiencies in our formulation. As a result, the system is indifferent to using energy directly from the solar panel or from the battery.

Ideally we would like the agent to learn a policy to ensure zero deviation from its initial battery level at the end of each episode. Low deviation are awarded large rewards and larger deviations are awarded with lower rewards.

4.2 SARSA(λ)

The reward at the end of an episode is used to update the Q-values of the state-action pairs according to Algorithm 1 [21]. The use of eligibility traces allows us to propagate the reward backwards to all the state-action pairs that contributed to it. Before the training, the Q-table is optimistically initialized to a high value $qinit$. At the start of each episode, the eligibility traces are reset and an action is selected randomly. The agent then determines its state and uses an ϵ-greedy policy to interact with its environment until the end of the episode. The reward it receives at the end of each episode is used to update the Q-table towards a better estimate.

4.3 Training Setup

We use historical data for training the agent. We take solar data of a year and apply SARSA(λ) learning for each day of that year for N number of iterations. The agent is trained in three phases. In the first phase of training, the initial battery level is initialized to some middle value. In the

ALGORITHM 1: SARSA(λ) algorithm

Initialize $Q(s, a) = qinit$ and $e(s, a) = 0$ for all s, a;
for *each episode* **do**
  Initialize s, a;
  for *each step of the episode* **do**
    Take action a, observe reward r and next state s';
    Choose next action a' from Q using ϵ-greedy policy;
    $\delta \leftarrow r + \gamma Q(s', a') - Q(s, a)$;
    $e(s, a) \leftarrow e(s, a) + 1$;
    for *all* (s, a) **do**
      $Q(s, a) \leftarrow Q(s, a) + \alpha\delta e(s, a)$;
      $e(s, a) = \gamma\lambda e(s, a)$;
    end
    $s \leftarrow s'; a \leftarrow a'$;
  end
end

second and third phase of training, this is initialized to high and low values respectively. This is done so that the agent is able to ensure ENO even when battery levels are not initialized at optimal levels. We observe the behavior of a trained agent by simulating it in an environment which it has not experienced during training. We allow the agent to learn as it interacts with the environment so that it can adapt and re-calibrate itself if need be.

5 SIMULATION METHODOLOGY

Here we describe the specifications of our system model and the parameters that are used during training and implementation.

5.1 System Setup

We base our system specification on a scaled up version up of a TMote Sky node powered by a 3.6V, 2200 mAh NiMh battery and a 6 W solar panel (220mm $\times$ 175mm). The node power consumption varies from approximately 100 mW to 20 mW depending upon whether the node is sensing, transmitting or receiving data. We scale up these values roughly by a factor of five. We chose this particular configuration simply for the sake of example. Any change in this system (for e.g. change in battery capacity or node power consumption) is accommodated for by the power manager due to its adaptive nature. This in fact is the essence of our approach - our power management policy is fluid enough to find a near-optimal policy for any realistic system specification without the need for any intervention by the user.

5.1.1 ***Energy Source:*** To simulate the solar energy harvested, we acquire global solar radiation data from the Japanese Meteorological Agency (JMA) website http://www.jma.go.jp. JMA provides hourly data on the global solar radiation several locations in Japan. As a result, we fix an epoch to be an hour long for our purpose. We use the solar radiation data to calculate the electrical energy generated by a solar panel.

5.1.2 ***Sensor Node:*** We consider a general sensor node that consumes energy varying from 100 mWh to 500 mWh during each epoch according to the specified duty cycle. We assume the power consumption remains constant within each epoch. The possible duty cycles are 20%, 40%, 60%, 80%

and 100%. We do not consider sensing latency inherent in sensor nodes for sake of simplicity. The power manager is indifferent to how the node allocates its power consumption.

5.1.3 ***Battery:*** Using the guidelines in [9], we calculate the battery size, B_{MAX}, and the optimal initial battery level, B_0. Using statistics for the year 2010, we arrive at $B_{MAX} = 40000mWh$ and $B_0 = 60\%$ of B_{MAX}. We assume an ideal battery for the sake of simplicity. Any inefficiencies of the node, solar cell and the battery can be lumped together as an increase in node power consumption in this context.

5.2 SARSA(λ) Parameters

5.2.1 Action Space. The action space, A, defines the actions that can be chosen by the agent. In our case, the action space is $A = \{20\%, 40\%, 60\%, 80\%, 100\%\}$.

5.2.2 State Definitions. The state of the system at some epoch t_k is given by

$$S_k = (S_{batt}(t_k), S_{dist}(t_k), S_{eharvest}(t_k), S_{day}(t_k)).$$

The state of the agent is determined from the actual values observed by the agent. Since these values are continuous in nature, it is necessary to discretize and define which values correspond to which states. We use the following to assign states from actual observed values.

$S_{batt}(t_k) \in \{S_{b1}, S_{b2}, S_{b3}\}$ gives information about whether the agent is in danger of depleting its battery or overcharging it. This is determined by the value of $e_{batt}(t_k)$ according to Table 1.

Table 1. $S_{batt}(t_k)$ Assignment

$S_{batt}(t_k)$	Range
S_{b1}	$e_{batt}(t_k) <$ 20% of B_{MAX}
S_{b2}	20% of $B_{MAX} \leq e_{batt}(t_k) <$ 80% of B_{MAX}
S_{b3}	80% of $B_{MAX} \leq e_{batt}(t_k) <$ 100% of B_{MAX}

$S_{dist}(t_k) \in \{S_{d1}, S_{d2}, ..S_{d40}\}$ gives information on how far the agent's operation is from ENO. This state is determined by the value of $e_{dist}(t_k)$ according to the following.

- $S_{dist}(t_k) \in \{S_{d22}, ..S_{d40}\}$ correspond to states in which $e_{dist}(t_k) < 0$ i.e. the battery is at a level lower than B_0.
- $S_{dist}(t_k) \in \{S_{d21}\}$ corresponds to the state of perfect energy neutrality i.e. $e_{dist}(t_k) = B_0 - e_{batt}(t_k) = 0$;
- $S_{dist}(t_k) \in \{S_{d1}, S_{d2}, ..S_{d20}\}$correspond to states in which $e_{dist}(t_k) > 0$ i.e. the battery is at a level higher than B_0.

Neighboring states are spaced at a distance of 1000 mWh from each other.

$S_{eharvest}(t_k) \in \{S_{e1}, S_{e2}...S_{e7}\}$ gives information about how much energy is being harvested at epoch (t_k). Its value is determined by $e_{harvest}(t_k)$ according to Table 2.

$S_{day}(t_k) \in \{S_{f1}, S_{f2}...S_{f6}\}$ gives the agent general information about what kind of weather it can expect during the day as shown in Table 3. In real world application, this information can be obtained from weather apps or websites. For our case, we differentiate each day into one of six different types according to the value of e_{day}, the total amount of energy harvested in that particular day. Since we are training on historical information, e_{day} can be easily calculated prior to the training by using Equation 7.

Table 2. $S_{eharvest}(t_k)$ Assignment

$S_{eharvest}$	Range
S_{e1}	$e_{harvest}(t_k) = 0mWh$
S_{e2}	$0mWh < e_{harvest}(t_k) \leq 100mWh$
S_{e3}	$100mWh < e_{harvest}(t_k) \leq 500mWh$
S_{e4}	$500mWh < e_{harvest}(t_k) \leq 1000mWh$
S_{e5}	$1000mWh < e_{harvest}(t_k) \leq 1500mWh$
S_{e6}	$1500mWh < e_{harvest}(t_k) \leq 2000mWh$
S_{e7}	$e_{harvest}(t_k) > 2000mWh$

$$e_{day} = \sum_{k=1}^{24} e_{harvest}(t_k) \tag{7}$$

Table 3. $S_{day}(t_k)$ Assignment

S_{day}	Weather	Range
S_{f1}	Very little sun	$e_{day}(t_k) < 2500mWh$
S_{f2}	Overcast	$2500mWh \leq e_{harvest}(t_k) < 5000mWh$
S_{f3}	Partly Cloudy	$5000mWh \leq e_{harvest}(t_k) < 8000mWh$
S_{f4}	Fair	$8000mWh \leq e_{harvest}(t_k) < 10000mWh$
S_{f5}	Sunny	$10000mWh \leq e_{harvest}(t_k) < 12000mWh$
S_{f6}	Very Sunny	$e_{harvest}(t_k) \geq 12000mWh$

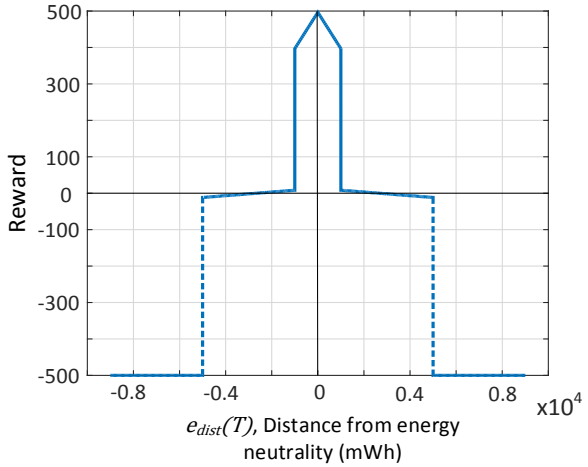

Fig. 2. Reward Function

5.2.3 Reward Function. The reward function in Figure 2 shows the relationship between the reward and the distance from energy neutrality at the end of the episode (day), $e_{dist}(T)$. Since

$e_{dist}(T)$ cannot always be exactly zero, we consider a margin of ±1000 mWh deviation as acceptable. The symmetry of the reward scheme is due to the fact that both positive and negative deviations from ENO have the same effect on the reward. During training, the rewards oscillate around some final value which introduces some noise in the reward. The sharp profile of the reward function allows the learning algorithm to arrive at true Q-values in spite of this noise. The gradual slope between ±1000 mWh and±5000 mWh deviation enables the agent to incrementally improve its performance without causing unstable oscillations during learning. This reward function is a novel contribution of our work. It is simple and intuitive and best reflects the ENO objective. Moreover, this reward function is independent of system model specifications and thus has a general scope of application.

5.3 Training Parameters

We execute Algorithm 1 with $\alpha = 0.001$, $\gamma = 0.8$ and $\lambda = 0.3$. As with most RL problems, these values were determined empirically rather than through mathematical methods. We evaluated the system with different combinations of the values of the hyper parameters and chose the combination that performed the best. The system is somewhat sensitive to the values of α and ϵ. Using a high values for α (learning rate) and ϵ causes large oscillations in Q-values during training. Hence, we chose smaller values but compensated with a larger number of iterations during learning.

Each epoch is an hour long with one episode consisting of 24 epochs. During training, the agent iterates $N(N = 10^6)$ times for each day of the year. For the three phases of training, we fix the initial battery level B_{INIT} at 60%, 80% and 20% of B_{MAX}.

We train our agent with weather data of Tokyo for the year 2010. We then observe its performance for the year 2011 in Tokyo and Wakkanai. Wakkanai lies in far north of Japan and experiences a drastically different weather than that of Tokyo. We use this change in location and climate to observe the node's adaptive behavior.

5.4 Evaluation Metrics

We refer to our power management policy as *SARSA Policy*. We compare our policy to a power management strategy mentioned in [9] referred here as *Offline Policy*. Offline Policy uses linear programming optimization methods with non-causal data on energy harvesting opportunities to determine the optimal duty cycles. The results of this method are presented here as an estimate of the upper limit of performance. This is not a practical method of power management because non-causal data on energy harvesting opportunities is not available in real life. The Offline Policy solutions have real continuous values and therefore to ensure fair comparison, the values of the duty cycles are rounded off to the nearest possible duty cycle of the system. As a result of this rounding off process, the Offline Policy also rarely every achieves perfect ENO. The Offline Policy uses an optimization window of one day (24 hours) to calculate the duty cycles. This also ensures fair comparison with our method because our SARSA(λ) agent is also trained in one-day episodes.

We express the battery levels, duty cycles and energy harvested in percentage of their maximum values. ENP are expressed as percentages of the maximum battery value B_{MAX}. We use root mean squared (RMS) values of ENP to compare performance between different policies.

6 RESULTS

6.1 Learning Convergence

Figure 3 shows the convergence of SARSA(λ) to its final optimal policy as the agent trains over a number of iterations. The graph shows the end of the day deviations from the initial optimal battery level, B_0 for each iteration during training on days 58 and 69 of the year 2010 (Tokyo).

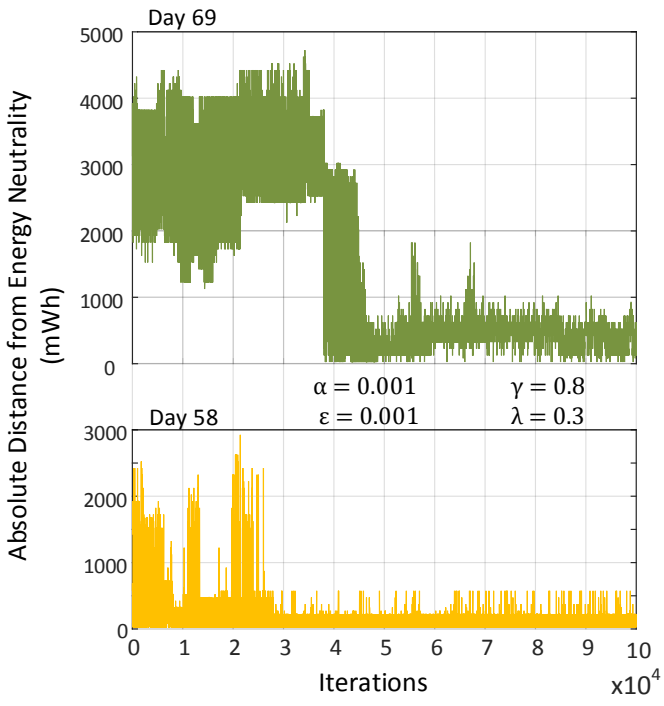

Fig. 3. Policy Convergence

We see that the agent is learning vigorously during the first 30000 to 40000 iterations. After 50000 iterations, it has figured out the optimal policy as is evidenced by the low deviation from energy neutrality in the figure. For the next 50000 iterations, it follows this policy greedily taking an occasional random action to see if it might lead to a better policy.

The fluctuations in the deviation from energy neutrality is the consequence of the limited number of possible discrete power consumption modes in the node. Since the harvested energy can take continuous values, it is highly unlikely that perfect energy neutral operation will ever be achieved. There will always be a small amount of error. We allow an error margin of ±1000 mWh (±2.5% of B_{MAX}) for the best case scenario. We tolerate up to ±5000 mWh (±12.5% of B_{MAX}) of deviation. This is also reflected in the reward function shown in Figure 2. Also, a small change in one of the node's actions may cause the deviation to jump between positive and negative values. This may explain the oscillation in deviation values as the agent converges to an optimal policy.

6.2 Energy Neutral Operation

Figure 4 shows the comparison of SARSA Policy with the Offline Policy for January 29, 2011 Tokyo. The *Constant Duty Cycle Policy* is determined by simply dividing the total energy harvested by 24 i.e. the total number of hours in one day. Of course, this constant duty cycle is not a realistic duty cycle that is achievable by the agent and is shown for comparison purposes only. The green dotted-dashed line indicates the initial level of the battery (B_0).

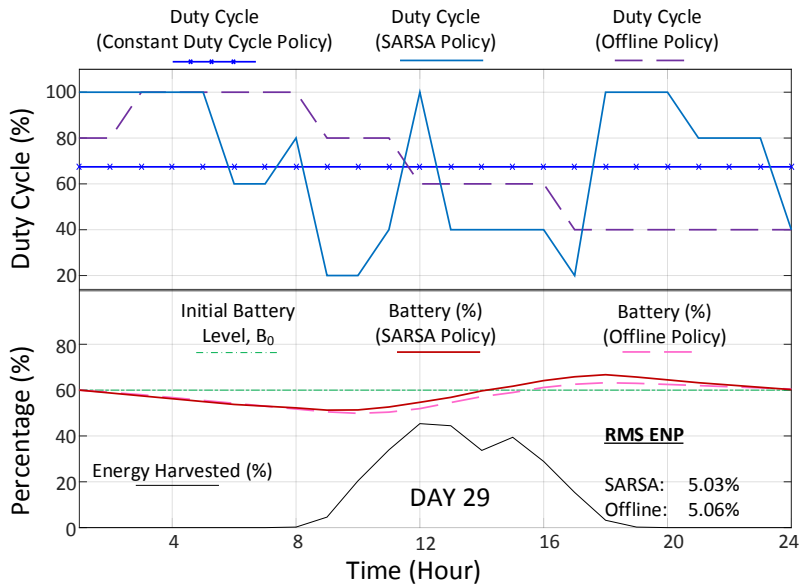

Fig. 4. SARSA and Offline Policy

It is worth noting that although the duty cycles of the Offline Policy and SARSA Policy differ from each other, they both have very similar performance. Both start the day off with 60% battery end with near about the same battery level. The ENP at the end of the day for SARSA Policy and Offline Policy was 491.875 mWh and 191.875 mWh respectively. SARSA Policy violated the energy neutrality by only 300 mWh (0.75% of B_{MAX}) more than that of Offline Policy. In fact, SARSA Policy shows slightly lesser RMS deviation (5.03%) as compared to the Offline Policy (5.06%). This shows that SARSA policy comes very close to optimal performance using only a general knowledge of the weather forecast.

As mentioned before, the Offline Policy optimizes its performance when considering one day at a time. In Figure 5, we compare SARSA Policy with the Offline Policy with a one-day window (shown in violet) as well as with an Offline Policy with a 30-day window (shown in blue) for 2011, Tokyo. We allow the duty cycles of the Offline Policy with a 30-day window to have continuous values between 10% and 100% and as a result, perfect energy neutrality is achieved using this policy.

We observe the battery profiles for different polices for a 30-day period. We observe that both SARSA Policy and one-day window Offline Policy have similar behavior and try and maintain the battery level at 60% of B_{MAX} at the end of every day. In contrast, the Offline Policy with a 30-day window deviates significantly from the optimal battery level during the middle of the 30-day period. The SARSA Policy can also be modified to optimize for a longer window. Such longer windows optimization techniques show better performance especially when the battery is nearing its limits. However they require more computation and exponentially longer training sessions.

Figure 5 also shows the battery profiles for two other policies - the constant duty cycle policy (shown in dotted blue) and a naïve battery-centric policy (shown in dashed orange). By averaging the non-causal information about the energy harvested over the 30 day period, we can determine the constant duty cycle to ensure ENO. This policy is not adaptive at all. Using a constant duty

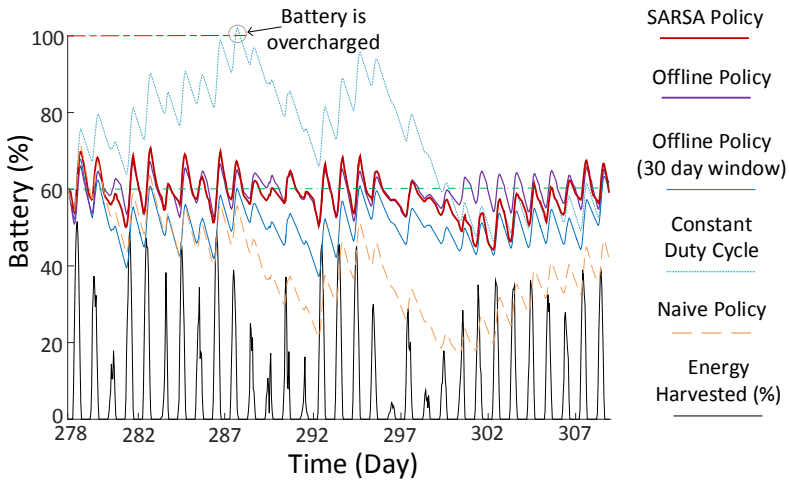

Fig. 5. Comparison of different policies

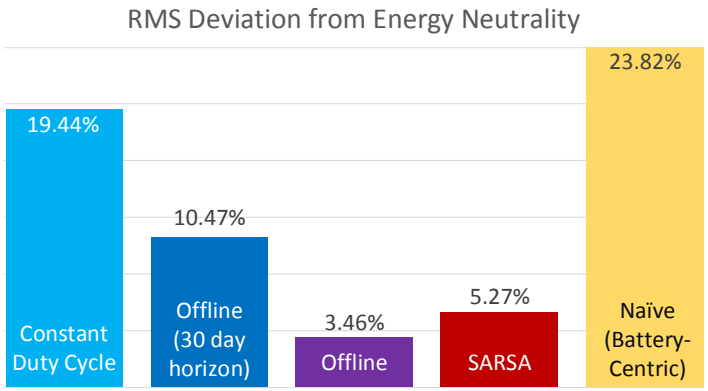

Fig. 6. Energy neutral performance of different policies

cycle policy may lead to battery overcharge/depletion and therefore wastage of energy and node failure. We can observe that some amount of energy is wasted due to overcharging during Day 286 as a result of this policy. The Naïve policy is the simplest adaptive policy. It is battery-centric in that the duty cycle is proportional to the battery reserve level. Higher battery drives the node with higher duty cycles and vice versa. While this policy is simple to implement, it is not very intelligent. For instance, on Day 288, the Naïve policy deviates quite far from the optimal battery level and drops to almost 20% on Day 292. If the days following Day 292 were not sunny enough, the battery could very well have been depleted. Figure 6 shows the RMS deviation from energy neutrality at the end of the 30 day period. We can see that the Naïve Policy suffers from the largest deviation at more than 23%. SARSA Policy and one-day window Offline Policy show very little deviation 3.46% and 5.27%. The constant duty cycle and 30-day horizon Offline policy exhibit perfect energy neutrality. However this comes at the cost of higher deviations as illustrated in Figure 5.

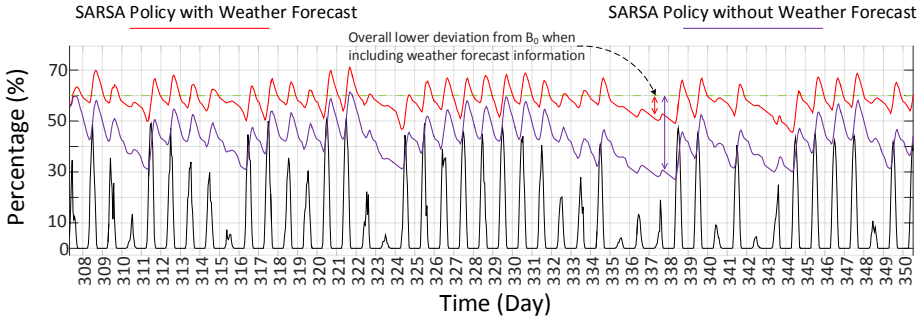

Fig. 7. Effect of weather forecast information

6.3 Effect of including weather forecast

In Figure 7 we compare the behavior of SARSA policies with and without forecast information from Day 308 to 324 (Wakkanai 2011). The policy that does not involve weather forecast in its decision making (violet) deviates more from the optimal battery level (shown in dotted-dashed green) than the policy that considers weather forecast information (shown in red). Both policies strive to achieve energy neutrality but the policy with weather forecast is more successful because it leverages on the weather forecast information to make better decisions. On the other hand, the weather agnostic policy uses a general policy for all weather types and this leads to lower performance in comparison.

6.4 Adaptation to seasonal changes

Figure 8 and 9 compare the SARSA Policy and Offline Policy for a week in spring and winter of Tokyo 2011. Figure 8 shows the plots for the week starting from February 27. The second and third day receive very little sunshine. However, starting from the fourth day, it gets a lot sunnier. We can see how SARSA adapts its strategy when it is anticipating a sunny or non-sunny day. We also observe that for the last three days of the week, both SARSA Policy and Offline Policy max out the duty cycle to 100% to be able to use all of the energy being harvested. In spite of their best efforts, the amount of energy that is harvested exceeds what can be consumed. This surplus is stored in the battery and is reflected in the rising battery level.

Conversely during the week (starting from Nov 29) in winter (Figure 9), the amount of energy harvested is not sufficient to sustain even the minimum duty cycle. During the first three days, both policies try to maximize their duty cycle as much as possible. However from the fourth day onwards, both policies fall back to the lowest duty cycle. The last day of the week is quite sunny and so both policies replenish their battery back to the optimum level. We see that SARSA has a similar RMS ENP compared to the Offline Policy during both spring and summer. This shows that the SARSA Policy is able to maintain near-optimal performance while still being able to adapt to seasonal changes.

6.5 Adaptation to climatic changes

To simulate a change in environment due to change in location, we observe the behavior of the node in Wakkanai, a place with a climate drastically different from which the node was originally trained on. Figure 10 shows the performance of both SARSA Policy and Offline Policy for a two-week period starting from February 4. We can see that both policies have similar behavior.

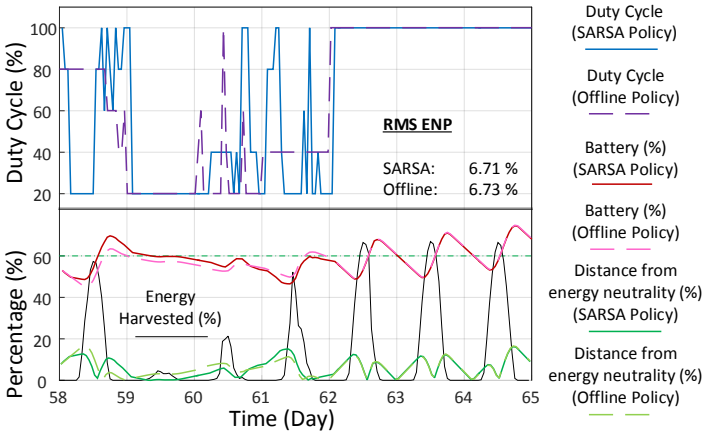

Fig. 8. Tokyo Spring 2011

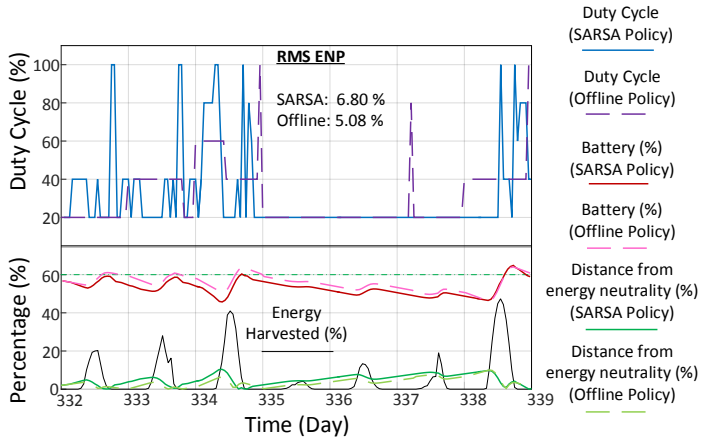

Fig. 9. Tokyo Winter 2011

The duty cycles corresponding to SARSA Policy have more variance than that of Offline Policy. This may be due to the fact that Offline policy has perfect prior knowledge of all energy harvesting data and so can optimize its policy better. SARSA Policy on the other hand does not have this information and instead has to decide on an action only after it observes the current harvested energy state. This may also explain why SARSA Policy has higher RMS ENP than Offline Policy.

We believe that SARSA Policy adapts quite well to Wakkanai environment because we have used distance from the energy neutrality as one of the state definition. Since the SARSA Policy is largely independent of the battery level (except when it is near the extreme limits), the strategy it uses during severe Tokyo winters also works for Wakkanai. Figure 11 is a color map that shows the deviation from energy neutrality for both policies in the year 2011. The top half shows the results for Tokyo whereas the bottom half is for Wakkanai. Each of the four color maps consists of 12 rows corresponding to each month in the year 2011. Each cell of a month row corresponds to a day in

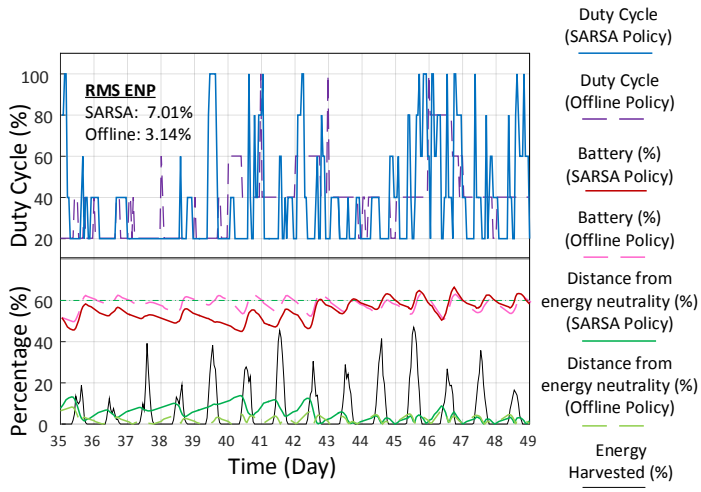

Fig. 10. Wakkanai Feb 2011

the month and its color indicates the deviation from energy neutrality at the end of that day. Blue cells indicate positive deviation i.e. more energy is harvested than that is consumed. Red indicates that more energy is consumed than that was harvested. Greener the cell, the lesser the deviation. The RMS ENP for each case is also shown alongside the figures.

The battery is initialized to the optimal value at the start of each day to get a fair idea of deviation from energy neutrality for each day. SARSA Policy is able to achieve ENO-Max operation in most cases. The RMS ENP for Offline Policy gives us an estimate of the best possible performance. We observe that SARSA Policy comes very close to achieving optimal performance by adapting to the changes in the environment.

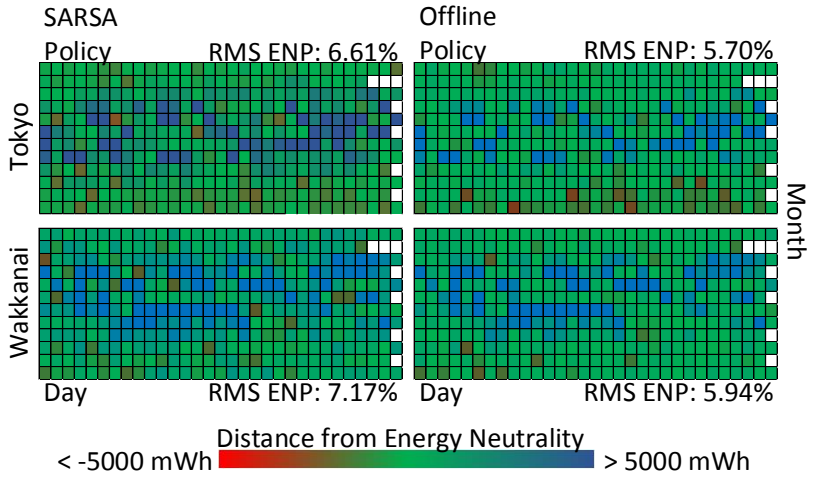

Fig. 11. Wakkanai vs Tokyo

6.6 Adaptation to battery degradation

To simulate the degradation of battery of the sensor node, we observe the behavior of SARSA Policy for 2011 Wakkanai weather with only half the original battery capacity. B_{MAX} is now reduced to 20000 mWh and $B_0 = 60\% of B_{MAX}$ becomes 12000. The SARSA Policy now tries to achieve ENO around this new B_0. Since the SARSA Policy gives greater weight to the distance of energy neutrality than the actual battery level, the policy is able to achieve energy neutral performance even when the battery capacity is halved. This is shown in Figure 12. The RMS ENP for Offline Policy again gives us an estimate of the upper limit in performance. We see that our policy comes very close to achieving this in spite of a drastic change in device parameters. When comparing with the Offline Policy, we see that SARSA Policy show a slightly reduced performance (as evidenced by fewer green cells) but has an overall satisfactory result.

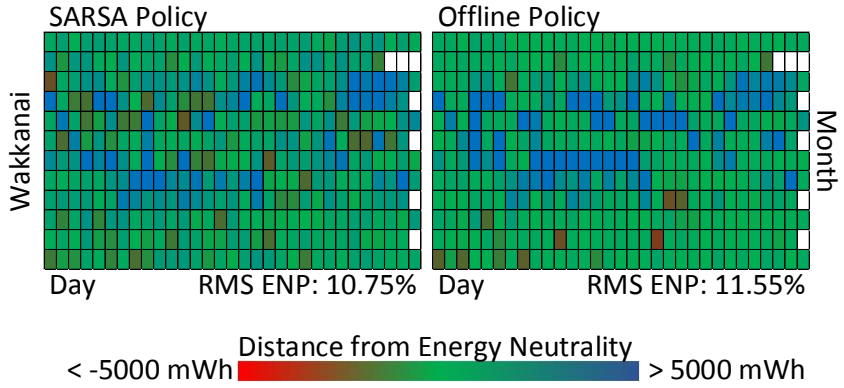

Fig. 12. Performance with half battery capacity

6.7 Adaptation to changes in device parameters

In real life application scenarios, the sensor node may have to accommodate for changes in its device parameters such as a decrease in energy harvesting efficiency and energy efficiency of the sensor node. Our proposed solution is able to adapt to such changes and learn to perform optimally in such scenarios. To simulate such a setting, we observe how SARSA policy learns to adapt to conditions when we

- halve the solar cell output
- increase the node's power consumption by 2.5 times (to simulate a degradation in the node's energy efficiency)

The sensor node (agent) has to be put in a "learning mode" so that it can adapt to these changes in working parameters. In this learning mode, the agent is allowed to train for 1000 iterations with $\alpha = 0.1$ and $\epsilon = 70$ in its new environment. The high learning rate and exploration rate allows it to quickly make changes in its learned Q-table. An additional advantage of this adaptive behavior is that the sensor node can make up for any initial calibration errors by the user. Even if the sensor node is assumed to be deployed in an environment it was not initially designed for, it can quickly adapt and work optimally. Thus, even though the simulation and real world applications may differ, our approach to power management control is able to accommodate these differences and still perform optimally.

Figure 13 shows the battery profiles for SARSA and Offline Policy with default device settings (without being put into the learning mode) for October 31, 2011. This is similar to the results in Figure 4. The only difference is that it is for a different date. We use this as the baseline performance.

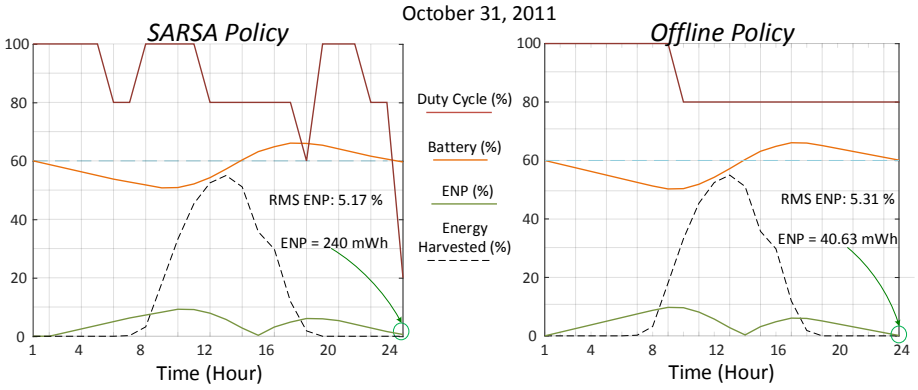

Fig. 13. Default Device Settings

Next, we observe the performance of the SARSA Policy when the solar panel output is reduced by half. This can be due to decrease of solar panel efficiency or a mismatch in design parameters during testing and implementation. The battery profiles for SARSA and Offline Policy is shown in Fig 14. We see that SARSA is able to achieve node level energy neutrality with very little battery deviation at the end of the day. In fact, in this particular case, SARSA policy has a lesser deviation from ENO than Offline Policy. Although the Offline Policy does represent the theoretical upper limit, the rounding off process involved can introduce some errors such as in this case.

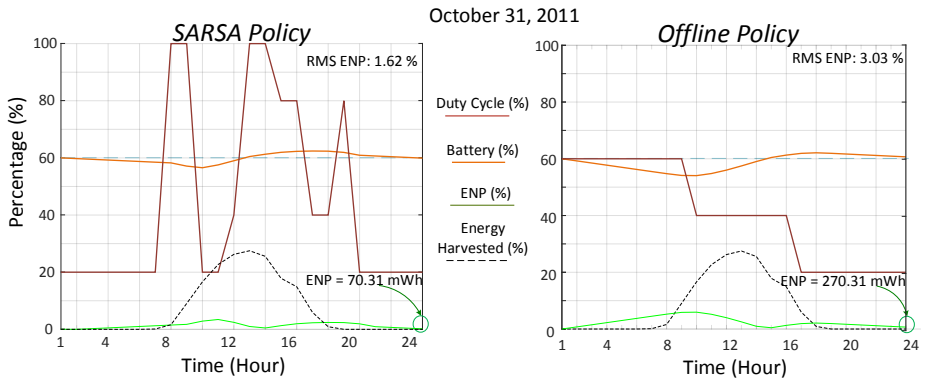

Fig. 14. Performance with half solar panel capacity

Finally, we consider the case when the sensor node consumes more power than it was initially assumed. This again maybe due to decrease in the node's working efficiency or mismatch in design and implementation. In Figure 15, we see that SARSA is able to achieve better performance than

Offline Policy in this case also. SARSA is achieves an ENP as low as 70.31 mWh at the end of the day.

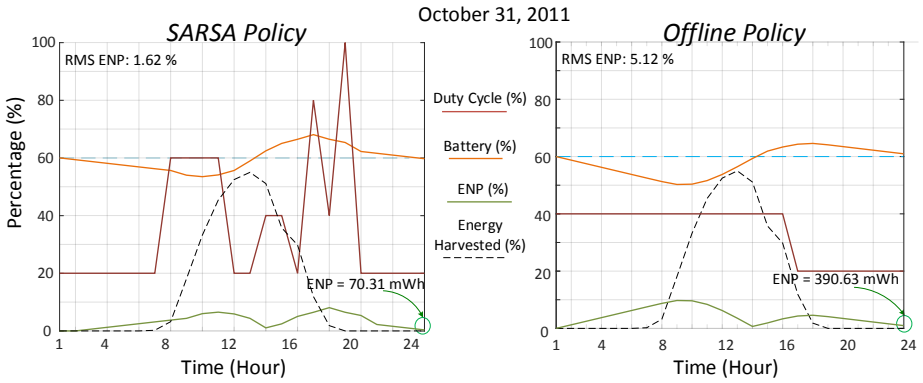

Fig. 15. Performance with increased node power consumption

7 CONCLUSION

We observe that our SARSA(λ) based RL approach achieves near perfect ENO making autonomous operation a possibility. We also see that our definitions of the state results in a highly adaptive behavior. Our proposed method is able to adapt to changes in weather, location (climate), battery degradation and device parameters which makes the sensor node robust in its operation. In addition, our state definition and general reward formulation scheme allows for general application of our power management method independent of the system it is being implemented in. We also show that inclusion of weather forecast information enhances the performance of the proposed scheme.

ACKNOWLEDGMENTS

This work was partially supported by JSPS KAKENHI Grant Number 16K12405. The first author acknowledges the Japanese Government (MEXT) Scholarship for his study in The University of Tokyo.

REFERENCES

[1] Luca Benini, Giuliano Castelli, Alberto Macii, and Riccardo Scarsi. 2001. Battery-driven dynamic power management. *IEEE Design & Test of Computers* 18, 2 (2001), 53–60.

[2] Pol Blasco and others. 2013. A learning theoretic approach to energy harvesting communication system optimization. *IEEE Tr. on Wireless Communications* 12, 4 (2013), 1872–1882.

[3] Wai Hong Ronald Chan and others. 2015. Adaptive duty cycling in sensor networks with energy harvesting using continuous-time Markov chain and fluid models. *IEEE Journal on Selected Areas in Communications* 33, 12 (2015), 2687–2700.

[4] Deniz Gunduz, Kostas Stamatiou, Nicolo Michelusi, and Michele Zorzi. 2014. Designing intelligent energy harvesting communication systems. *IEEE Communications Magazine* 52, 1 (2014), 210–216.

[5] Jason Hsu and others. 2006. Adaptive duty cycling for energy harvesting systems. In *Proc. of the 2006 ISLPED*. 180–185.

[6] Roy Chaoming Hsu and others. 2014. A Reinforcement Learning-Based ToD Provisioning Dynamic Power Management for Sustainable Operation of Energy Harvesting Wireless Sensor Node. *IEEE Tr. on Emerging Topics in Computing* 2, 2 (2014), 181–191.

[7] Roy Chaoming Hsu and others. 2015. Dynamic energy management of energy harvesting wireless sensor nodes using fuzzy inference system with reinforcement learning. In *IEEE 13th INDIN*. 116–120.

[8] Xiaofan Jiang, Joseph Polastre, and David Culler. 2005. Perpetual environmentally powered sensor networks. In *Proceedings of the 4th international symposium on Information processing in sensor networks*. IEEE Press, 65.
[9] Aman Kansal and others. 2007. Power management in energy harvesting sensor networks. *ACM Tr. on Embedded Computing Systems* 6, 4 (2007), 32.
[10] Aman Kansal, Dunny Potter, and Mani B Srivastava. 2004. Performance aware tasking for environmentally powered sensor networks. *ACM SIGMETRICS Performance Evaluation Review* 32, 1 (2004), 223–234.
[11] Junaid Ahmed Khan and others. 2015. Energy management in wireless sensor networks: a survey. *Computers & Electrical Engineering* 41 (2015), 159–176.
[12] Cheng-Ting Liu and Roy Chaoming Hsu. 2011. Dynamic power management utilizing reinforcement learning with fuzzy reward for energy harvesting wireless sensor nodes. In *37th Annual Conference on IEEE Industrial Electronics Society*. 2365–2369.
[13] S Danish Maqbool and others. 2011. Analysis of adaptability of Reinforcement Learning approach. In *IEEE 14th INMIC*. 45–49.
[14] Nicolò Michelusi and others. 2013. Energy management policies for harvesting-based wireless sensor devices with battery degradation. *IEEE Tr. on Communications* 61, 12 (2013), 4934–4947.
[15] Andrea Ortiz and others. 2016. Reinforcement learning for energy harvesting point-to-point communications. In *2016 IEEE International Conference on Communications*. 1–6.
[16] Vijay Raghunathan, Aman Kansal, Jason Hsu, Jonathan Friedman, and Mani Srivastava. 2005. Design considerations for solar energy harvesting wireless embedded systems. In *Proceedings of the 4th international symposium on Information processing in sensor networks*. IEEE Press, 64.
[17] Luigi Rucco and others. 2013. A bird's eye view on reinforcement learning approaches for power management in WSNs. In *6th WMNC*. 1–8.
[18] Navin Sharma and others. 2010. Cloudy computing: Leveraging weather forecasts in energy harvesting sensor systems. In *7th IEEE SECON*. 1–9.
[19] Sujesha Sudevalayam and Purushottam Kulkarni. 2011. Energy harvesting sensor nodes: Survey and implications. *IEEE Communications Surveys & Tutorials* 13, 3 (2011), 443–461.
[20] Srikanth Sundaresan and others. 2009. Event-driven adaptive duty-cycling in sensor networks. *International Journal of Sensor Networks* 6, 2 (2009), 89–100.
[21] Richard S Sutton and others. 1992. Reinforcement learning is direct adaptive optimal control. *IEEE Control Systems* 12, 2 (1992), 19–22.
[22] Emre Ünsal, Taner Akkan, L Özlem Akkan, and Yalçın Çebi. 2016. Power management for Wireless Sensor Networks in underground mining. In *Signal Processing and Communication Application Conference (SIU), 2016 24th*. IEEE, 1053–1056.
[23] Christopher M Vigorito and others. 2007. Adaptive control of duty cycling in energy-harvesting wireless sensor networks. In *4th IEEE Communications Society Conference on Sensor, Mesh and Ad Hoc Communications and Networks*. 21–30.

Received April 2017; revised June 2017; accepted July 2017