A Fast and Optimal Smart Home Energy Management System: State-Space Approximate Dynamic Programming

ZHIHENG ZHAO1, (Student Member, IEEE), AND
CHANAKA KEERTHISINGHE2,3, (Member, IEEE)
1School of Electrical Engineering and Telecommunications, University of New South Wales, Sydney, NSW 2052, Australia
2Department of Electrical and Computer Engineering, University of Washington, Seattle, WA 98195, USA
3Systems Integration Testbed, Washington Clean Energy Testbeds, Seattle, WA 98105, USA
Corresponding author: Chanaka Keerthisinghe (chanakak@ieee.org)

ABSTRACT Dynamic programming (DP) can be used to generate the optimal schedules of a smart home energy management system (SHEMS), however, it is computationally difficult because we have to loop over all the possible states, decisions and outcomes. This paper proposes a novel state-space approximate dynamic programming (SS-ADP) approach to quickly solve a SHEMS problem but with similar solutions as DP. The state-space approximations are made using a hierarchical approach, which involves clustering and machine learning. The proposed SS-ADP can generate the day-ahead value functions quickly without compromising the solution quality because it only loops over the necessary state-space. Our simulation results showed that the solutions from the SS-ADP approach are within 0.8% of the optimal DP solutions but saves the computational time by at least 20%. The paper also presents a fast real-time control strategy under uncertainty using the Bellman optimality condition and long short-term memory recurrent neural networks (LSTM-RNN). The Bellman equation uses the day-ahead value function from the SS-ADP and the instantaneous contribution function to make fast real-time decisions. The instantaneous contribution is calculated using the PV and load predicted using LSTM-RNN, which performs significantly better than the widely used persistence method.

INDEX TERMS PV-storage systems, smart home energy management, state-space approximate dynamic programming, machine learning, long short-term memory recurrent neural networks.

I. INTRODUCTION
A Smart home is an automated residential building that uses distributed energy resources for managing energy consumption and providing suitable levels of comfort to its inhabitants. The number of smart homes with photovoltaic (PV) and battery storage systems has increased dramatically in some parts of USA, China, Australia, New Zealand and Europe, in response to rising electricity costs, government incentives, decreasing installation costs, and growing concerns about climate change.

In order to maximize the benefits of PV-storage systems, residential energy users will use a smart home energy management system (SHEMS) to schedule their energy use.

A SHEMS consists of a day-ahead planning and real-time decision making stages as depicted in Fig. 1. The day-ahead planning involves solving a stochastic sequential decision making problem using the day-ahead predictions of PV generation, load and electricity prices.

This underlying optimization problem can be solved using deterministic and stochastic mixed-integer linear programming (MILP) [1]–[9], particle swarm optimization (PSO) [10]–[14], dynamic programming (DP) [9], [15]–[17], approximate dynamic programming (ADP) [15], [16], [18]–[23] and policy function approximations using machine learning [24]. Solving the day-ahead planning problem using MILP and PSO means that we have to either resolve a difficult optimization problem during the real-time decision making process when uncertainties regarding PV, load and price arise, which will require a high computational
power. Moreover, these off-the-shelf solvers might not result in a solution for some constraints. For example, the SHEMS parameters that are set by the user can result in constraints to the optimization problem that have no possible solution.

On the other hand DP and ADP generate value functions and value function approximations, respectively, in the day-ahead planning stage, which lets the user make fast real-time decisions using Bellman optimality condition, which is a much faster process compared to having to solve a difficult optimization problem. However, DP is computationally difficult due to dimensionalities of state, decision and outcome spaces and forward ADP results in slightly lower quality solutions. Note that DP results in close-to-optimum solutions when the state, decision, and outcome spaces are finely discretized, and moreover, we can consider non-linear constraints such as battery degradation and efficiencies without any additional computational burden.

A SHEMS should be computationally inexpensive to be integrated into an existing smart meter or a raspberry pi board. Implementing a SHEMS in a cloud would mean that the computational speed would become less important, however, since the computational time of DP increases exponentially when we increase the number of devices [16], it would be beneficial either way to keep the computational cost to a minimum level.

Given these insights, we first propose a hierarchical approach for state-space approximate dynamic programming (SS-ADP) to generate the value functions quickly but without affecting the solution quality. The trick here is to approximate the state-space of the problem so the DP only needs to loop over the necessary states. The approach proceeds as follows: First, a historical dataset of battery state-of-charge (SoC) profiles are generated under different PV generation, electrical load and price values using a deterministic solver. Second, cluster this data into different groups according to the daily battery SoC pattern. Third, identify the correct cluster before the day-ahead optimization using machine learning so the modified state-space can be generated. A machine learning based state-space approximations have been proposed in [25] to solve an energy management problem of a power plant, which uses artificial neural networks for predicting the state-space.

The real-time decision-making stage according to the Bellman optimality condition is as follows: First the value function at a given time-step from the SS-ADP is combined with the instantaneous contribution function, which is generated using the real-time forecasts of PV and load. Second, the decisions corresponding to the maximum value of this combined function is selected as the optimal decisions. This paper also proposes to forecast PV and load using long short-term memory recurrent neural networks (LSTM-RNN), which is significantly better than the widely used persistence forecasts.

The PV output and electrical load profiles used in this paper are from data [26] collected during the Smart Grid Smart City (SGSC) project by Ausgrid in New South Wales, Australia, which investigated the benefits and costs of implementing a range of smart grid technologies in Australian households [27].

Section II presents the problem formulation. The proposed SS-ADP approach is in Section III. Modeling procedure of the day-ahead SHEMS problem is presented in Section IV. Section V estimates the real-time PV and load using LSTM-RNN. The real-time control strategy using the Bellman optimality condition is explained in Section VI. Simulation results and the discussion is in Section VII. Section VIII concludes the paper.

II. SMART HOME ENERGY MANAGEMENT PROBLEM

In Section II-A, we present the general formulation of the sequential stochastic optimization problem and then Section II-B formulates our SHEMS problem in this form.

A. GENERAL SEQUENTIAL STOCHASTIC OPTIMISATION PROBLEM

A sequential stochastic optimization problem consists of:

- A sequence of time-steps, $K = \{1 \ldots k \ldots K\}$, where $K$ and $k$ represent the total number of time-steps and a particular time-step in the decision horizon, respectively.
- A set of non-controllable inputs, $J = \{1 \ldots j \ldots J\}$, where each $j$ is represented using:
  - A state variable, $s^j_k \in S$.
  - A random variable, $\omega^j_k \in \Omega$, that captures perturbations or exogenous information.
- A set of controllable devices, $I = \{1 \ldots i \ldots I\}$, where each $i$ is represented using:
  - A state variable, $s^i_k \in S$;
  - A decision variable, $x^i_k \in X$ for every control action;
  - Constraints for the control and state variables;
A transition function $s^i_{k+1} = s^M(s^i_k, x^i_k, o^i_k)$ describes the evolution of a state from $k$ to $k+1$. Here $s^M(.)$ is the system model that consists of the operational constraints of controllable device $i$, such as power flow limits, efficiencies and losses. Given this, let $s_k = [s^i_k, \ldots, s^d_{k-1}]$, $x_k = [x^i_k, \ldots, x^d_k]^T$, and $o_k = [o^i_k, \ldots, o^d_k]^T$. The state variables contain the information that is necessary and sufficient to make the decisions and compute rewards, costs and transitions. The compact form of the transition functions is given as $s_{k+1} = s^M(s_k, x_k, o_k)$. Note that we only need transition functions for the controllable devices and the combined random variables vector of the non-controllable inputs is given by $o_k$ (without a superscript).

- An objective function:

$$F = \mathbb{E} \left\{ \sum_{k=1}^{K} C_k(s_k, x_k, o_k) \right\}, \quad (1)$$

where $C_k(s_k, x_k, o_k)$ is the contribution (i.e. reward or cost of energy, or a discomfort penalty) incurred at time-step $k$, which accumulates over time.

**B. INSTANTIATION**

The aim of the SHEMS is to minimize energy costs over a decision horizon. In general, stochastic optimization techniques solve this sequential stochastic optimization problem before the beginning of each day, using either a daily or a two day decision horizon. Here we consider a smart home with a PV and battery, as depicted in Fig. 2 for convenience, two day decision horizon. Here we consider a smart home as state variables and the variation as random variables. Electricity price, electrical load and PV output in the optimization. We model the stochastic variables using their mean (i.e. reward or cost of energy, or a discomfort penalty) incurred at time-step $k$, which accumulates over time. The non-controllable inputs are the electrical load, PV output, and electricity price, which are represented using $\omega_k$.

The energy balance constraint is given by:

$$s^d_k + \omega^d_k = s^p_k + \omega^p_k - \mu^b s^b_k + \gamma^b_k, \quad (2)$$

where $\mu^b$ is the efficiency of the inverter (note that the efficiency is 1/\mu^e when the battery is charging); $\mu^p$ is the efficiency of the battery action corresponding to either charging or discharging; and $x^p_k$ is the electrical grid power. The charge rate of the battery is constrained by the maximum charge rate $s^b_k = \gamma^c_k$ (i.e. charging is positive) and discharge rate of the battery is constrained by the maximum discharge rate $x^b_k = \gamma^d_k$ (i.e. discharging is negative). The energy stored in the battery should be within the limits $s^b_{k,\min} \leq s^b_k \leq s^b_{k,\max}$ according to the manufactures instructions.

- Transition functions govern how the state variables evolve over time. The battery SoC, denoted $s^b_k \in [s^b_{k,\min}, s^b_{k,\max}]$, progresses by:

$$s^b_{k+1} = \left(1 - l^b(s^b_k)\right) \left(s^b_k - s^b_{k,\min} + \mu^b s^b_k + \gamma^b_k\right), \quad (3)$$

where $l^b(s^b_k)$ models the self-discharging process of the battery. The battery SoC transition function is a non-linear function of state. The discharge efficiency of the battery, efficiency of the inverter and battery degradation constraints are non-linear and are given in Fig. 3 and [16]. The number of battery cycles that a battery can have decreases exponentially as the depth-of-discharge increases [28]. The remaining device characteristics are as follows: the charging efficiency of the battery is $\mu^{\text{eff}} = 1$; the battery capacity is 10 kWh; and the maximum charge and discharge rates of the battery are 4 kWh. The self-discharging losses of the battery per half-hour is 0.005 $s^b_{k,\max}$. The battery is operated between 10% to 100% SoC.

The optimal policy, $\pi^*$, is a choice of action for each state $\pi: S \rightarrow X$, that minimizes the expected sum of future costs over the decision horizon; that is:

$$F^{\pi^*} = \min_{\pi} \mathbb{E} \left\{ \sum_{k=0}^{K} C_k(s_k, \pi(s_k), o_k) \right\}, \quad (4)$$
induction, a procedure called value iteration, and then an optimal policy is extracted from the value function by selecting a minimum value action for each state. This is the key functional point of difference between DP and stochastic MILP. DP enable us to plan offline by generating value functions for every time-step. Once we have the value functions, we can make faster online solutions using (7) (more details are towards the end of this section). Note that a value function at a given time-step consists of the expected future cost from all the states. This process of mapping states and actions is not possible with stochastic MILP.

An illustration of a deterministic DP using a simplified model of a battery storage is shown in Fig. 4. At every time-step, there are three battery SoC states (i.e. highest, middle, and lowest) and three possible battery actions that results in different instantaneous costs. At the last time-step, k = K, the expected future cost from the desired state, s_k = M, is zero, while the other two states are penalised with a large cost. This is an important step that allows us to control the end-of-day battery SoC (discussed in Section V). The expected future cost at every possible state is calculated using (7), which is the minimum of the combined instantaneous cost that results from the decision that we take and the expected future cost from the state we end up at the next time-step. In Fig 4, instantaneous cost is on the edges of the lines while the expected future cost is below the states. An optimal policy is extracted from the value functions by selecting a minimum value action for each state using (7). For example, from s_1^b, if we take the optimal decision to go to s_2^b = L then the total combined cost of 10 consists of a instantaneous cost of 2 and a expected future cost of 8. Even though the expected future cost of 7 from s_2^b = M is lower than the expected future cost from s_2^b = L, the instantaneous cost that takes us there is 4 so the total combined cost is 11. Given this, the expected future cost of following the optimal policy from s_1^b is 10 and at time-step 2 we will be at s_2^b = L.

![FIGURE 3. Characteristics of the battery and inverter: (a) Inverter efficiency; (b) Battery discharge efficiency; (c) Number of cycles before reaching 80% end-of-life vs. depth-of-discharge; and (d) Calendar aging with varying SoC at 25°C.](image)

where \( C_k(s_k, x_k, \omega_k) \) is the cost incurred at a given time-step, which is given by:

\[
C_k(s_k, x_k, \omega_k) = s_k^p \left( s_k^{pv} + \omega_k^{pv} - \mu_i \mu_x x_k^b - s_k^{d,e} - \omega_k^{d,e} \right).
\]

(5)

Here the problem is formulated as an optimization of the expected contribution because the contribution is generally a random variable due to the effect of \( \omega_k \). In all the SHEMSs, we obtain the battery decisions \( x_k^b \) depending on the state variables \( s_k = [x_k^b, x_k^{pv}, x_k^{d,e}, s_k] \), and realisations of random variables \( \omega_k = [\omega_k^{pv}, \omega_k^{d,e}] \) at each time-step.

### III. A HIERARCHICAL APPROACH FOR STATE-SPACE APPROXIMATE DYNAMIC PROGRAMMING

This section first explains DP and then presents the hierarchical approach used to make state-space approximations that improve the computational burden of DP.

#### A. DYNAMIC PROGRAMMING (DP)

The problem in (4) is easily cast as a Markov decision process (MDP) due to the separable objective function and Markov property of the transition functions. Given this, DP solves the MDP form of (4) by computing a value function \( V^\pi(s_k) \). This is the expected future cost of following a policy, \( \pi \), starting in state, \( s_k \), and is given by:

\[
V^\pi(s_k) = \sum_{s' \in S} \mathbb{P}(s'|s_k, \pi(s_k), \omega_k) \left[ C(s_k, \pi(s_k), s') + V^\pi(s') \right].
\]

(6)

An optimal policy, \( \pi^* \), is one that minimizes (4), and which also satisfies Bellman’s optimality condition:

\[
V^*_{k+1}(s_{k+1}) = \min_{\pi^*} \left( C_k(s_k, \pi(s_k)) + \mathbb{E} \left( V^*_{k+1}(s'_{k+1}) | s_k \right) \right).
\]

(7)

The expression in (7) is typically computed using backward induction, a procedure called value iteration, and then an optimal policy is extracted from the value function by selecting a minimum value action for each state. This is the key functional point of difference between DP and stochastic MILP. DP enable us to plan offline by generating value functions for every time-step. Once we have the value functions, we can make faster online solutions using (7) (more details are towards the end of this section). Note that a value function at a given time-step consists of the expected future cost from all the states. This process of mapping states and actions is not possible with stochastic MILP.

![FIGURE 4. A deterministic DP example using a battery storage, where expected future cost is calculated using (7). The instantaneous contributions from the battery decisions are on the edges of the lines while the expected future cost is below the states. The optimal policy satisfies (4), which is obtained using (7).](image)
There are several reasons to prefer DP over MILP. First, DP produces close-to-optimal solutions when the value functions obtained during the offline planning phase are from finely discretized state, action and outcome spaces. Second, in practical applications, the SHEMS can make real-time decisions using the policy implied by (7). This means that at each time-step, the optimal decision from the current state can be executed. Note that (7) is a simple linear program at each time-step so is computationally feasible using existing smart meters. This stands in contrast to a stochastic MILP formulation, which would involve solving the entire stochastic MILP program, which is computationally difficult even for the offline planning. Third, we can always obtain a solution with DP regardless of the constraints and the inputs while MILP fails to find a solution when the constraints are not satisfied.

However, the required computation to generate value functions using DP grows exponentially with the size of the state, action and outcome spaces. This paper overcomes the computational burden of DP using state-space approximations, while maintaining the benefits of DP.

**B. AN HIERARCHICAL APPROACH FOR STATE-SPACE APPROXIMATIONS**

This section presents the proposed hierarchical approach used to make state-space approximations. The aim is to only loop over the necessary states, so the solution quality stays the same as DP. Here only the state-space reduction of battery SoC is considered, however, the approach can be used for other controllable distributed energy resources (DER).

The hierarchical approach shown in Fig. 1(b) proceeds as follows:

- First historical battery SoC profiles over a day are generated by solving a deterministic optimization problem by assuming perfect foresight. This can be done using any optimization technique as this is only needed to be done once.
- Second cluster these historical battery SoC profiles into a range of patterns using a k-means algorithm. Attributes used are the average battery SoC values at certain periods of the day.

Next, in the day-ahead planning stage, use the day-ahead electrical load and PV forecast to generate the day-ahead battery SoC profile by solving the SHEMS problem. Here any optimization technique can be used because the battery SoC values are only needed to be accurate enough to correctly identify the cluster that it belongs. This classification is done using k-nearest neighbors algorithm. Finally, this chosen cluster is used to obtain the modified state-space for our stochastic DP approach where a finer discretization is used for the battery SoC values that are in the chosen cluster.

Fig. 5 shows the upper and lower limits of the battery SoC over a day generated using the proposed approach. This area is finely descretized so the SS-ADP results in the same battery SoC profile as DP.

**IV. MODELING THE DAY-AHEAD SHEMS PROBLEM**

This section models the day-ahead SHEMS problem using the proposed SS-ADP and DP.

The stochastic inputs to the day-ahead optimization (formulated in Section 2 I-II), PV output and electrical load, are estimated using the clustering and kernel regression-based approach proposed in [29]. In summary, the approach proceeds as follows: First historical electrical load and PV generation data are clustered according to the load pattern and total daily PV output, respectively. The PV generation clusters represent day-types such as sunny days, cloudy days etc. The electrical load clusters are defined in a similar way but also using the time of the energy use as an attribute. The median values in each cluster is the predicted profile. Second, the probability density functions within these clusters are determined using kernel regression. Finally, we can use a less accurate forecast to choose the correct PV and load clusters for a given day. An example of these models is shown in Fig. 6. The median profile of a chosen cluster is the predicted profile for a deterministic optimization while the probability models are used for the stochastic optimization.
The day-ahead value function approximations in Fig. 7(a) are generated using the proposed SS-ADP approach. The example shows the expected future contribution for following the optimal policy from time-steps $k = 2$, $k = 11$, $k = 30$ and $k = 35$ vs. battery SoC. Note that the expected future contribution increases with the battery SoC but the instantaneous contribution either decreases or stays the same with the increasing battery charge rate. The instantaneous contribution for battery charging using solar energy is zero.

![FIGURE 7. (a) Value functions and (b) Instantaneous contribution function.](image)

Given the high uncertainties associated with the day-ahead PV and load forecasts, the schedules generated from the day-ahead optimization have large errors. Moreover, solving the entire optimization problem at every time-step would require expensive computational power. To overcome this, [16] proposes to use the day-ahead value functions from DP and ADP to generate real-time decisions according to the Bellman optimality condition (more details in Section VI). In order to accurately generate the instantaneous contribution function, SHEMS needs to forecast the PV and load at the next time-step.

V. REAL-TIME FORECASTS OF PV AND LOAD USING LSTM-RNN

This section proceeds as follows: First a brief review of existing solution techniques used to forecast the real-time PV and load (i.e. 30 minutes ahead forecast) is presented. Second the chosen LSTM-RNN is explained. Third and fourth sub-sections present the modeling and results, respectively.

A. BRIEF REVIEW OF EXISTING SOLUTION TECHNIQUES

The underlying PV and load forecasting problem is a time series prediction problem. First we review PV forecasting techniques, which can be categorized into physical models and data-driven models. Physical models use numerical weather prediction, which shows good performance for forecast horizons from several hours up to six days [30]. Data-driven models can be further divided into statistical and machine learning models. Statistical models include auto-regressive integrated moving average, coupled auto-regressive and dynamic system, Lasso, and Markov models. Machine learning models include support vector machine, feed-forward neural networks, and RNN such as LSTM networks [31]–[33]. These approaches can be further subdivided according to the type of input features that are used to train the model. A forecasting model that uses only a target time-series as an input feature (solar irradiance in this case) is referred to as a nonlinear auto-regressive (NAR) model. On the other hand, if a model uses additional exogenous inputs, such as temperature and humidity, it is referred to as a nonlinear auto-regressive with exogenous inputs (NARX) model. According to [31], [32], LSTM-RNN performs the best at forecasting PV generation.

Similar approaches have been used to forecast residential load such as feed forward neural networks, support vector machine, and RNN such as LSTM networks and gate-recurrent units [34], [35]. According to [34], [35], LSTM-RNN performs the best at forecasting residential load. Given this, this paper focuses on LSTM-RNN for forecasting both the 30 minute ahead PV generation and electrical load.

B. LONG SHORT-TERM MEMORY RECURRENT NEURAL NETWORKS

LSTM networks are a special type of RNN that are capable of learning long-term dependencies. Fig. 8 illustrates the RNN and Fig. 9 the LSTM cells [31]. Note that the variables defined are only applicable to this section.

![FIGURE 8. Recurrent neural network. The circles represent the input, hidden and output layers and the weights are above the lines. The arrows show the direction of forward propagation. Note that the variables defined are only applicable to Section V-B.](image)

A RNN has a feedback connection where the output depends on the current input to the network, and the previous inputs, outputs, and/or hidden states of the network, as depicted in Fig. 8. Given an input sequence $x = (x^1, \ldots, x^T)$, RNNs compute the hidden vector $h = (h^1, \ldots, h^T)$, and output vector sequence $y = (y^1, \ldots, y^T)$ by iterating (8) and implementing the calculation shown in (9).

$$h_t = \mathcal{H}(W_{sh} \cdot x_t + W_{hh} \cdot h_{t-1} + b_h),$$  \hspace{1cm} (8)$$y_t = W_{hy} \cdot h_t + b_y,$$  \hspace{1cm} (9)$$$$\text{where } \mathcal{H} \text{ is the activation function of the hidden layer. } W_{sh}, W_{hh} \text{ and } W_{hy} \text{ are the weight matrices of the input-hidden,}$$
The hidden and output bias vectors are hidden-hidden, and hidden-output connections, respectively. FIGURE 9. Long short-term memory cell.

The hidden and output bias vectors are $b_h$ and $b_o$.

The error gradients of RNNs that are trained using back-propagation through time can accumulate during an update and result in very large gradients. These gradients result in large updates to the network weights, and in turn, an unstable network. At an extreme, the weights can become so large as to overflow and result in non-computable values. LSTM networks have been developed to overcome the exploding back-propagated gradients of RNNs by providing explicit memory to the network. LSTM units are building units for layers of RNNs. A typical LSTM unit consists of an input gate, forget gate, output gate, and a cell unit.

The operations of the LSTM unit can be described as follows. The most important component is the cell state $c_t$, which serves as a memory and remembers values over an arbitrary time interval. The input gate $i_t$, forget gate $f_t$, and output gate $o_t$ control the flow of information into and out of the cell and has the same size as the hidden vector $h_t$.

The forget gate $f_t$ outputs a 0 or 1 for each number in cell state to decide what information we want to put out from the previous cell state $c_{t-1}$, according to:

$$ f_t = \sigma \left( W_f \cdot x_t + W_f \cdot h_{t-1} + b_f \right), \quad (10) $$

where $\sigma$ is the sigmoid activation function. $W_f$ and $b_f$ are the weight matrix and bias of the forget gate, respectively.

Similarly, the input gate output $i_t$ decides the new input information that should accumulate in the memory cell:

$$ i_t = \sigma \left( W_i \cdot x_t + W_i \cdot h_{t-1} + b_i \right), \quad (11) $$

where $W_I$ and $b_i$ are the weight matrix and bias of the input gate, respectively.

The LSTM cell state is then updated as follows, but with conditional self-loop weights $W_c$ and $b_c$:

$$ c_t = f_t \cdot c_{t-1} + i_t \cdot \text{tanh} \left( W_c \cdot x_t + W_c \cdot h_{t-1} + b_c \right). \quad (12) $$

The output hidden state $h_t$ of the LSTM cell depends on the cell state $c_t$ and the output gate $o_t$:

$$ h_t = o_t \cdot \text{tanh}(c_t). \quad (13) $$

The output gate $o_t$ is calculated using:

$$ o_t = \sigma \left( W_o \cdot x_t + W_o \cdot h_t + b_o \right), \quad (14) $$

where $W_o$ and $b_o$ are the weight matrix and bias of the output gate. Note that the hidden state $h_t$ can be shut off via the output gate $o_t$, which uses a sigmoid activation function.

C. MODELING

PV and load forecasting using machine learning involves three steps: (1) Preparing the dataset, (2) Training, and (3) Predicting. Steps one and two are done day-ahead while step three is done in real-time. Historical data are separated into training and testing sets. Note that only one feature is used for predicting PV output and load (i.e. we implemented a NAR model). Two years of data are used for training while one year is used for testing.

D. RESULTS

The real-time 30 minute-ahead forecasts of the PV output and electrical load generated using LSTM-RNN are compared with the widely used persistence approach where the forecast for the next time-step is the current value.

Table 1 compares the two approaches using the root-mean-square error (RMSE). The PV generation forecasts from LSTM-RNN are significantly better than the persistence approach while the electrical load forecasts are only slightly better. The LSTM-RNN forecasts of the electrical load can be further improved if we consider the number of inhabitants and their behavioral patterns as features of the LSTM model (i.e. a NARX model). Unfortunately, the dataset used in this paper only consists of the aggregated load profiles so we were unable to show the complete benefits of using LSTM-RNN for electrical load prediction.

### TABLE 1. Yearly PV and load one time-step ahead forecasting error for LSTM-RNN and persistence approaches. Improvements as a percentage in the parenthesis.

|            | PV        | Load      |
|------------|-----------|-----------|
| LSTM-RNN   | 0.0535 (27.41%) | 0.1917 (62.5%) |
| Persistence|           | 0.1929    |
| Household A| 0.1213 (44.28%) | 0.5147 (2.94%) |
| Persistence|           | 0.2177 (53.03%) |

VI. REAL-TIME DECISION MAKING USING BELLMAN OPTIMALITY CONDITION

This section explains the real-time decision making process using the Bellman equation (7). In the real-time decision making process, the real-time forecasts of the PV and electrical load values are used to update the instantaneous contribution function (example in Fig. 7(b)) according to (5). Next, this instantaneous contribution function is added to the day-ahead value function of the next step that was generated during the offline planning stage. Note that there will be a value function
for every time-step in a day. Finally, the optimum battery decision for the next time-step is the battery decision corresponding to the maximum value of this combined function (i.e. Bellman optimality condition).

**VII. SIMULATION RESULTS AND DISCUSSION**

This section presents the simulation results from the proposed SS-ADP approach and DP in Table 2 for two households over a year. The yearly electricity cost from the proposed SS-ADP approach can be reduced by 45.85% and 39.72%, respectively by having a PV-battery system that is controlled by SS-ADP. The number of deep battery cycles that greatly affects the battery lifetime is also less as shown in Fig. 10. The benefits of a SHEMS has been very well studied so our focus here is to propose this computationally efficient algorithm that provides similar quality solutions as the DP.

**TABLE 2. Yearly Simulation Results. Improvements as a percentage in the parenthesis.**

|                       | Household 1 | Household 2 |
|-----------------------|-------------|-------------|
| PV generation (MWh)   | 2.89        | 5.31        |
| PV system size (kWp)  | 2.22        | 3.78        |
| Yearly electrical load (MWh) | 4.38 | 12.85 |
| Battery capacity (kWh) | 10          | 10          |
| Benchmark cost ($)     | 558.03      | 1588.2      |
| DP cost ($)            | 299.82      | 951.46      |
| (46.27%)               | (40.09%)    |             |
| SS-ADP cost ($)        | 302.15      | 957.39      |
| (45.85%)               | (39.72%)    |             |
| DP computational time per day (s) | 151.97 | 411.99 |
| SS-ADP computational time per day (s) | 119.52 | 300.43 |
| (21.35%)               | (27%)       |             |

The total yearly electricity cost of Household 1 and 2 can be reduced by 45.85% and 39.72%, respectively by having a PV-battery system that is controlled by SS-ADP. The number of deep battery cycles that greatly affects the battery lifetime is also less as shown in Fig. 10. The benefits of a SHEMS has been very well studied so our focus here is to propose this computationally efficient algorithm that provides similar quality solutions as the DP.

**FIGURE 10. Number of battery cycles used in a year vs. the depth-of-discharge for (a) Household A and (b) Household B.**

**REFERENCES**

[1] Z. Chen, L. Wu, and Y. Fu, “Real-time price-based demand response management for residential appliances via stochastic optimization and robust optimization,” *IEEE Trans. Smart Grid*, vol. 3, no. 4, pp. 1822–1831, Dec. 2012.

[2] M. C. Bozchalui, S. A. Hashmi, H. Hassen, C. A. Canizares, and K. Bhattacharya, “Optimal operation of residential energy hubs in smart grids,” *IEEE Trans. Smart Grid*, vol. 3, no. 4, pp. 1755–1766, Dec. 2012.

[3] T. Hubert and S. Grijalva, “Realizing smart grid benefits requires energy optimization algorithms at residential level,” in *Proc. ISGT*, Jan. 2011, pp. 1–8.

[4] Z. Zhu, J. Tang, S. Lambotharan, W. H. Chin, and Z. Fan, “An integer linear programming and game theory based optimization for demand-side management in smart grid,” in *Proc. IEEE GLOBECOM Workshops (GC Wkshps)*, Dec. 2011, pp. 1205–1210.

[5] F. De Angelis, M. Boaro, D. Fuselli, S. Squartini, F. Piazza, and Q. Wei, “Optimal home energy management under dynamic electrical and thermal constraints,” *IEEE Trans. Ind. Informat.*, vol. 9, no. 3, pp. 1518–1527, Aug. 2013.

[6] J. Wang, Z. Sun, Y. Zhou, and J. Dai, “Optimal dispatching model of smart home energy management system,” in *Proceedings of IEEE PES Innov. Smart Grid Technol. (ISGT Europe)*, Oct. 2012, pp. 1–8.

[7] H. Joumaa, G. De-Oliviera, S. Ploix, and M. Jacomino, “Energy management problem in dwellings: Combining centralized and distributed solving approaches,” in *Proceedings of IEEE PES Innov. Smart Grid Technol. (ISGT Europe)*, Oct. 2012, pp. 1–8.

[8] K. C. Sou, J. Weimer, H. Sandberg, and K. H. Johansson, “Scheduling smart home appliances using mixed integer linear programming,” in *Proc. IEEE Conf. Decis. Control Eur. Control Conf. (CCECC)*, Dec. 2011, pp. 5144–5149.

[9] C. Keerthisinghe, G. Verbič, and A. C. Chapman, “Evaluation of a multi-stage stochastic optimisation framework for energy management of residential PV-storage systems,” in *Proc. Australs: Universities Power Eng. Conf. (AUPEC)*, Sep. 2014, pp. 1–6.

[10] M. A. Pedrasa, E. D. Spooner, and I. F. MacGill, “Improved energy services provision through the intelligent control of distributed energy resources,” in *Proc. IEEE Bucharest PowerTech*, Jun. 2009, pp. 1–8.

[11] M. A. Pedrasa, T. D. Spooner, and I. F. MacGill, “Coordinated scheduling of residential distributed energy resources,” in *Proc. IEEE Trans. Smart Grid*, vol. 1, no. 2, pp. 134–143, Sep. 2010.

ACKNOWLEDGMENT

(Zhiheng Zhao and Chanaka Keerthisinghe contributed equally to this work.)
[13] M. Pedrasa, T. D. Spooner, and I. F. MacGill, “Scheduling of demand side resources using binary particle swarm optimization,” *IEEE Trans. Power Syst.*, vol. 24, no. 3, pp. 1173–1181, Aug. 2009.

[14] L. Wang, Z. Wang, and R. Yang, “Intelligent multiagent control system for energy and comfort management in smart and sustainable buildings,” *IEEE Trans. Smart Grid*, vol. 3, no. 2, pp. 605–617, Jun. 2012.

[15] C. Keerthisinghe, G. Verbic, and A. C. Chapman, “Energy management of PV-storage systems: ADP approach with temporal difference learning,” in *Proc. Power Syst. Comput. Conf. (PSCC)*, Jun. 2016, pp. 1–7.

[16] C. Keerthisinghe, G. Verbic, and A. C. Chapman, “A fast technique for smart home management: ADP with temporal difference learning,” *IEEE Trans. Smart Grid*, vol. 9, no. 4, pp. 3291–3303, Jul. 2018.

[17] D. P. Bertsekas, *Dynamic Programming and Optimal Control*, vol. 1, 3rd ed. Belmont, MA, USA: Athena Scientific, 2005.

[18] R. N. Anderson, A. Boulanger, W. B. Powell, and W. Scott, “Adaptive stochastic control for the smart grid,” *IEEE Trans. Power Syst.*, vol. 19, no. 6, pp. 1098–1115, Jun. 2004.

[19] H. S. W. S. Warren B. Powell, A. George, “Smart: A stochastic multiscale model for the analysis of energy resources, technology and policy,” Dept. Oper. Res, Financial Eng, Princeton Univ., Princeton, NJ, USA, Tech. Rep. 08544, Aug. 2010. [Online]. Available: http://www.castlelab.princeton.edu/Papers/SMARTAugust112010.pdf

[20] W. B. Powell, “Approximate dynamic programming for large-scale resource allocation problems,” Princeton Univ., Princeton, NJ, USA, Tech. Rep. 08544, 2005.

[21] J. Si, A. Barto, W. Powell, and D. Wunsch, *Handbook of Learning and Approximate Dynamic Programming*. Hoboken, NJ, USA: Wiley, Jan. 2004, doi: 10.10109/9780470544785.

[22] W. B. Powell, *Approximate Dynamic Programming-Solving the Curses of Dimensionality*. Hoboken, NJ, USA: Wiley, 2007.

[23] P. Samadi, H. Mohsenian-Rad, V. W. S. Wong, and R. Schober, “Real-time pricing for demand response based on stochastic approximation,” *IEEE Trans. Smart Grid*, vol. 5, no. 2, pp. 789–798, Mar. 2014.

[24] C. Keerthisinghe, A. C. Chapman, and G. Verbic, “Energy management of PV-storage systems: Policy approximations using machine learning,” *IEEE Trans. Ind. Informat.*, vol. 15, no. 1, pp. 257–265, Jan. 2019.

[25] C. Keerthisinghe, H. Sun, Y. Takaguchi, D. Nikoski, and H. Hashimoto, “Machine learning based state-space approximate dynamic programming approach for energy and reserve management of power plants,” in *Proc. IEEE Innov. Smart Grid Technol. Asia (ISGT Asia)*, May 2018, pp. 669–674.

[26] E. L. Ratnam, S. R. Weller, C. M. Kellett, and A. T. Murray, “Residential load and rooftop PV generation: An Australian distribution network dataset,” *International J. Sustain. Energy*, vol. 36, no. 8, pp. 787–806, 2017.

[27] Ausgrid, NSW, Australia. Solar Home Electricity Data. Accessed: May 20, 2020. [Online]. Available: http://www.ausgrid.com.au/Common/About-us/Corporate-information/Data-to-share/Solar-home-electricity-data.aspx

[28] Z. Zhao, C. Keerthisinghe: Fast and Optimal SHEMS: SS-ADP. ZHIHENG ZHAO (Student Member, IEEE) received the B.E. (Hons.) and M.E. degrees in electrical and electronic engineering from The University of Sydney, Sydney, NSW, Australia, in 2012 and 2016, respectively. He is currently pursuing the Ph.D. degree in electrical engineering with the University of New South Wales, Sydney. He has industrial work experience with Veolia Water Technologies and Solutions and the National Information Communication Technology Association, Sydney. His research interests include demand response, energy management in smart homes, thermal energy storage and implementation, green buildings, and the Internet of Energy.

CHANKA KEERTHISINGHE (Member, IEEE) was born in Sri Lanka. He received the B.E. (Hons.) and M.E. degrees in electrical and electronic engineering from The University of Auckland, Auckland, New Zealand, in 2011 and 2014, respectively, and the Ph.D. degree in electrical engineering from The University of Sydney, Sydney, NSW, Australia, in 2017. He is currently the Lead Electrical Engineer with the Systems Integration Testbed, Washington Clean Energy Testbeds. He is also a Postdoctoral Researcher with the Department of Electrical and Computer Engineering, University of Washington, Seattle, WA, USA. He has work experience with Mitsubishi Electric Research Laboratories, Cambridge, MA, USA. He holds a U.S. patent and publications in microgrids, energy management in smart homes, wireless power transfer, disaster response, electricity markets, stochastic optimization, and machine learning.

[29] S. Xie, X. Hu, S. Qi, X. Tang, K. Lang, Z. Xin, and J. Brighton, “Model predictive energy management for plug-in hybrid electric vehicles considering optimal battery depth of discharge,” *Energy*, vol. 173, pp. 667–678, Apr. 2019. [Online]. Available: http://www.sciencedirect.com/science/article/pii/

[30] C. Keerthisinghe, A. C. Chapman, and G. Verbic, “PV and demand models for a Markov decision process formulation of the home energy management problem,” *IEEE Trans. Ind. Electron.*, vol. 66, no. 2, pp. 1424–1433, Feb. 2019.

[31] M. Husein and I.-Y. Chung, “Day-ahead solar irradiance forecasting for microgrids using a long short-term memory recurrent neural network: A deep learning approach,” *Energies*, vol. 12, no. 10, p. 1856, May 2019. [Online]. Available: https://ideas.repec.org/a/gam/jener/v12y2019i10p1856-d231480.html

[32] Y. Yu, J. Cao, and J. Zhu, “An LSTM short-term solar irradiance forecasting under complicated weather conditions,” *IEEE Access*, vol. 7, pp. 145651–145666, 2019.

[33] J. Brownlee. (2019). How to Develop Multi-Step LSTM Time Series Forecasting Models for Power Usage. [Online]. Available: https://machinelearningmastery.com/how-to-develop-lstm-models-for-multi-step-time-series-forecasting-of-household-power-consumption/

[34] T. Hossen, A. S. Nair, R. A. Chinnathambi, and P. Ranganathan, “Residential load forecasting using deep neural networks (DNN),” in *Proc. North Amer. Power Syst. Nap.* Apr. 2018, pp. 1–5.

[35] B. Asare-Bediako, W. L. Kling, and P. F. Ribeiro, “Day-ahead residential load forecasting with artificial neural networks using smart meter data,” in *Proc. IEEE Grenoble Conf.*, Jun. 2013, pp. 1–6.