Micro-scale Smart Grid Optimization

Nathan Kowahl, and Anthony Kuh, *Member, IEEE*

***Abstract*—The adoption of smart grid technologies will allow for more distributed generation of energy and for residential and commercial users of electricity to make intelligent decisions about energy usage. In previous research by Livengood and Larsen [1], a stochastic dynamic programming problem is formulated for a micro-scale smart grid system. A mathematical model of energy usage is developed where the goal is to optimize a finite horizon cost function reflecting both the cost of electricity and comfort/lifestyle. This paper extends this work by assuming key models and forecasts are unknown and implicitly learned via the softmax algorithm with neighborhood updating. The algorithm implements approximate dynamic programming with a goal of reducing dependancies on models and forecasting while achieving good performance. Simulations are conducted using the softmax algorithm showing that the solution approaches the optimal dynamic programming algorithm solution.**

I. Introduction

Implementation of the Smart Grid will change the way we use and generate electricity. Some features include variable energy costs (depending on time of day), distributed generation, and end users making decisions about how they control energy usage. Smart grid systems may be classified as those which operate at the macro (national, state, regional) level, and those which operate at the micro (home, business, and neighborhood) level. At the macro-scale the smart grid system is concerned with utility security and reliability and two-way integration of renewable resources. At the micro-scale a smart grid system may be employed to effectively manage a home or business owner's utility load in a manner which allows him or her to save money by making the most of the resources available at any given time. Micro-scale smart grid systems will play a critical role in managing costs associated with the variable utility pricing models foreseen in the future. Smart grid systems (and variable pricing models) aim to 'shave the peaks, and fill in the valleys' [1] of the demand on electricity load. The role of the system is to make intelligent decisions based on the utility pricing scheme, availability of local resources (wind, solar, and battery power), and local demand for electricity load. The discussion and results presented here will be concerned with the development of a micro-scale smart grid optimization system, herein referred to as the smart system.

The remainder of this work includes a discussion of the modeling framework developed by Livengood and Larsen [1], the fundamentals of optimal decision making through dynamic programming and their results, and finally discussion and results of a more generalized solution to the optimization problem.

Nathan Kowahl is with the Department of Electrical Engineering, University of Hawaii at Manoa, Honolulu, Hawaii 96822; (email: nathanrk@hawaii.edu).

Anthony Kuh is with the Department of Electrical Engineering, University of Hawaii at Manoa, Honolulu, Hawaii 96822; (email: kuh@hawaii.edu).

II. Smart System Optimization

Decisions must have an outcome, and that outcome must have an associated cost. For the smart system the decisions revolve around efficient delegation of electrical load resources. Some examples of load delegation are storing electricity from the utility in the early morning hours to avoid peak charges later in the day, and turning down the air heating/cooling systems when utility prices are high. Costs for these decisions must be considered as both the current cost associated with, say, pulling electricity from the utility to store in the battery, as well as the future (potentially negative) costs associated with being able to use this resource at a later time. Optimal decision making becomes a balancing act between current and future costs. For the smart system considered here, future costs are heavily dependent on a number of variables, many of which are random in nature. The relationships between decisions and costs are critical to optimal decision making, and must be estimated through models or experience.

A. The Energy Box Model

In April of 2009, Livengood and Larson released their paper *The Energy Box: Locally Automated Optimal Control of Residential Electricity Usage* [1]. The paper outlines the dynamics of a variable pricing utility model, and the opportunity for locally controlled utility load management. A key aspect of the paper is the development of a state space model in which various utility management schemes may be tested. The model contains aspects foreseen in the future of utility management. These include renewable sources, such as wind power, and battery storage, which might be found in an electric vehicle, as well as variable utility pricing.

The smart system has several options for decreasing the costs associated with utility use including load scheduling, and load shedding. An example of load scheduling may include waiting to start the clothes dryer, or dishwasher. An example of load shedding may include turning off electric water heaters, and reducing cooling temperatures in refrigerators and air conditioning systems. Livengood and Larsen's system model focuses on load shedding associated with an air cooling system in the home. Decision making revolves around the air conditioner set point, and buying or selling of electricity from or to the grid via battery storage. The model provides several variables describing the state of the system which are used to make decisions. These include the time of day, indoor temperature, outdoor temperature, uncontrolled

978-1-4244-8126-2/10/$26.00 ©2010 IEEE

utility load in the home, utility price, wind speed (which determines wind power available), and battery charge level (see figure 1).

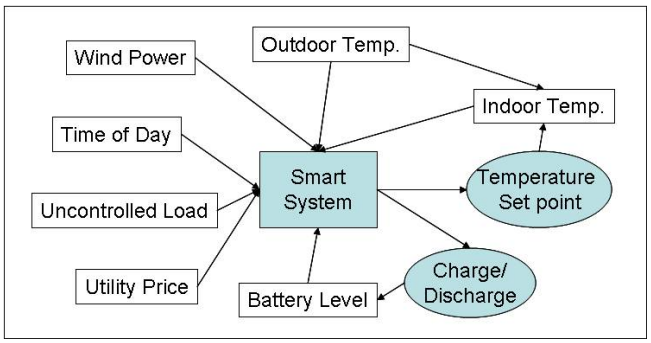

Fig. 1. The smart system attempts to optimize comfort and dollar costs associated with thermostat setting. After observing the indoor and outdoor temperatures, available wind power, time of day, uncontrolled load in the home or business, utility price, and battery level, the system outputs a directive to charge or discharge the battery, and set the thermostat.

The paper proposes a stochastic dynamic programming optimization solution for minimizing both dollar and comfort costs associated with utility use and conservation. As shown in figure 2 the stochastic dynamic programming optimization routine takes in a state forecast for the current episode and generates optimal decisions. These decisions in turn affect the costs and state transitions. Cost and state transition models are updated, and new forecasts feed the next round of decision making.

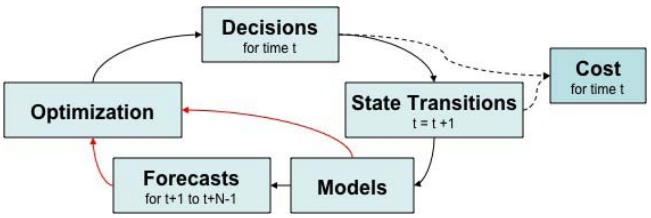

Fig. 2. In [1] decisions are optimized with stochastic dynamic programming given a forecast for the upcoming episode ($t = 1$ to $t = N - 1$). These decisions then affect the state transitions and associated costs. The next stage of states updates cost and transition models for forecasting, and the process repeats for each stage.

Forecasts are required for the outdoor temperature, utility price, wind speed, and uncontrolled load. Models are required for indoor temperature, battery level, and cost.

B. Stochastic Dynamic Programming

Dynamic programming is a method for system optimization performed by breaking the problem down into small, simple steps [4]. The fundamentals of dynamic programming revolve around six concepts: decisions, states, stages, the cost function, state transition rules, and decision rules. The *decisions* are made according to the *state* at each *stage* of the process. *States* evolve according to *state transition rules* which are a function of the *state* and the *decision* at the previous stage. The state evolution will dictate the cost as evaluated by the *cost function*. Decision rules are formulated for each state in order to minimize the cost function which is a function of current and future *stage* costs. Solutions to the dynamic programming problem may be found using the principle of optimality as developed by Richard Bellman [4]. For every feasible state a decision rule must be established which appropriately balances current and future costs. When the problem has a finite horizon (or final stage) it is most convenient to start from the final stage and work backwards, tallying future costs and optimal decisions as the problem grows.

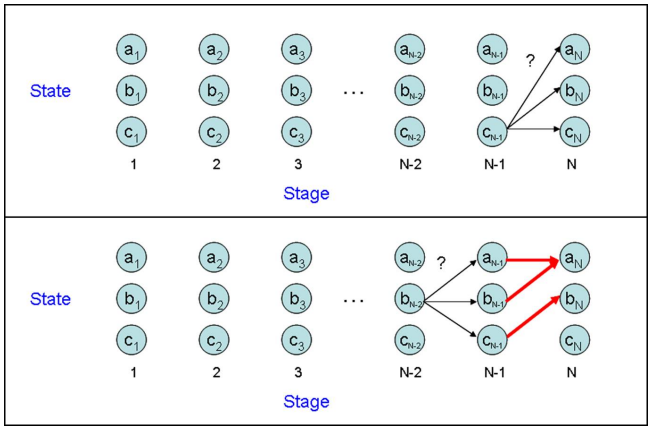

Fig. 3. The principle of optimality states that we may work backwards through stages in a dynamic programming problem in order to efficiently calculate future costs and optimal decisions for every state and stage. Potential decisions are shown in black, while calculated optimal decisions are shown in red. In the lower panel decisions are considered for state b_{N-2} given the optimal decisions determined in the upper panel.

Figure 3 depicts the calculation of optimal decisions and associated costs for an N stage horizon using the principle of optimality. In the top panel of figure 3 the last decision stage is considered which requires no knowledge of future costs. In the lower panel of figure 3 decisions are considered given current and future expected costs resulting from optimization at stage $N - 1$. Given a state and a decision, the state transition rules dictate the state evolution. The state transitions in figure 3 may be deterministic or stochastic. In stochastic problems state transitions are probabilistic. Thus, when calculating future costs for the system one must take the expected value of multiple costs in order to account for probabilistic state transitions.

$$U_t^* = \arg\min_{U_t} E\left[J\left(X_t, U_t\right) + J^*\left(X_{t+1}\right)\right] \quad (1)$$

$$X_{t+1} = P_{U_t}\left(X_t\right) \quad (2)$$

Eqs. (1) - (2) provide a generalized mathematical expression for optimal decision making where U_t^* is the optimal decision at stage t. Here X_t dontes the state at stage t, U_t is a potential decision at stage t, $J\left(\cdot\right)$ is the current stage cost and $J^*\left(\cdot\right)$ is the future stage cost under optimal decision making, considering state transition probabilities given by $P_{U_t}\left(\cdot\right)$. For a stochastic state transition the future stage cost

will be the sum of optimal stage costs weighted by the state transition probabilities.

C. Smart System State and Cost Models

For simulation purposes, state models are established to provide forecast input, and cost output to the optimization scheme. Each of the state models are presented in the following section. Given that optimal decision making implies determining an appropriate action for every state, care must be taken to reduce the total number of states available at any given stage. Given a forecast, many states are deterministic $\pm$ a single discrete noise term as in [1], reducing the number of possible values for the given state variable at the stage to 3. Even with this consideration there are 24 different time states, 15 states associated with indoor temperatures (integers between 68 and 82), 3 states for outdoor temperature, 3 states for the uncontrolled load, 3 states for the grid price, 16 states for the wind speed, and 5 states for the battery level. In total this generates 777,600 states. For every state there are 7 temperature set-point options, and 2 or 3 battery (dis)charging options. In order to accurately implement a stochastic dynamic programming solution 14,152,320 costs must be calculated. This is a very large number, and underlines a significant issue even given the severe discretization of the state models.

It should be noted here that this work is focused on the optimization of a smart system, not on the real-world accuracy of the particular underlying state and cost models. While state and cost models are constructed in an attempt to mimic real-world scenarios, it is anticipated that true realizations will vary extensively in different environments. The solution methods presented here are intended to work on a wide range of state and cost models. Specific attention is not given to the units of the different state variables.

1) Outdoor Temperature: The outdoor temperature is modeled as a sinusoid with additive noise. Figure 4 depicts a realization of the outdoor temperature model for one 24 hour episode.

$$\text{forecast} = \cos\left(\frac{(t - shift)\,\pi}{12}\right)$$

$$\text{actual} = \begin{cases} forecast - \Delta_{OT} & w.p.0.2 \\ forecast & w.p.0.6 \\ forecast + \Delta_{OT} & w.p.0.2 \end{cases}$$

$$\Delta_{OT} = 1$$

2) Indoor Temperature: The models for the indoor temperature are from [1] and [5]. The indoor temperature is deterministic given outdoor temperature, and set-point. Notice that the outdoor temperature and set-point also determine the load of the air conditioning unit which is capable of heating and cooling to meet the set-point. Figure 5 depicts the evolution of the indoor temperature given no air conditioning input. The indoor temperature basically lags that of the outdoor temperature owing to inefficiencies in insulation.

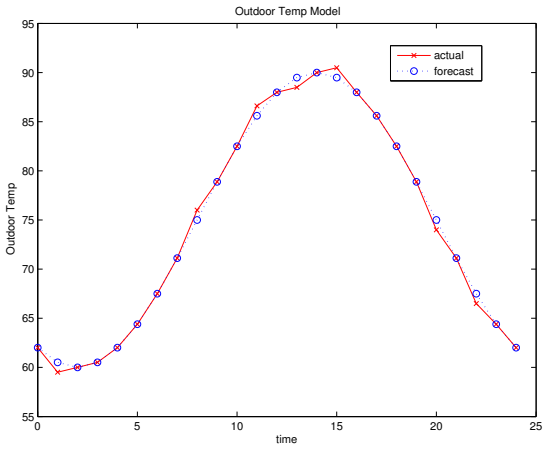

Fig. 4. Outdoor temperature model. Here $shift = 2$.

$$newTemp = \epsilon \times oldTemp + (1-\epsilon)\,(outdoorTemp - \gamma \times ACLoad)$$

$$ACLoad = \left| \frac{outdoorTemp - \frac{setTemp - \epsilon \times oldTemp}{1-\epsilon}}{\gamma} \right|$$

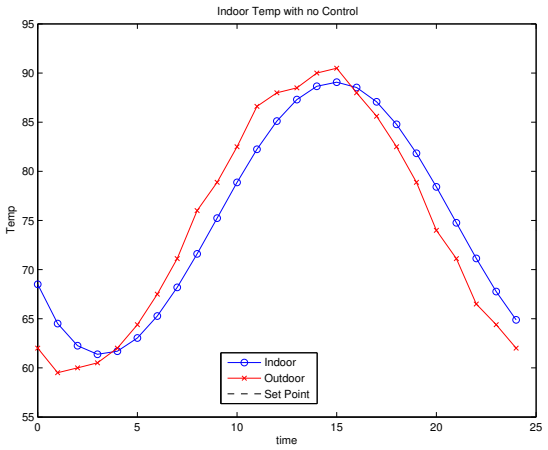

Fig. 5. Indoor temperature is determined by the outdoor temperature and the set-point. Here $\epsilon = 0.5$ and $\gamma = 0.2$

3) Uncontrolled Load: The uncontrolled load is modeled as a piece-wise sinusoid with multiplicative noise. This model is intended to reflect the user demand for electricity load throughout the waking hours, while maintaining a low background load during sleep. Figure 6 depicts a realization of the uncontrolled load model for a 24 hour episode.

$$forecast = \begin{cases} low & t < start \\ \sin\left(\frac{(t-start)\pi}{end-start}\right)(high-low)+low & otherwise \\ low & t > end \end{cases}$$

$$actual = \begin{cases} forecast\,(1-\Delta_{UL}) & w.p.0.2 \\ forecast & w.p.0.6 \\ forecast\,(1+\Delta_{UL}) & w.p.0.2 \end{cases}$$

$$\Delta_{UL} = 0.05$$

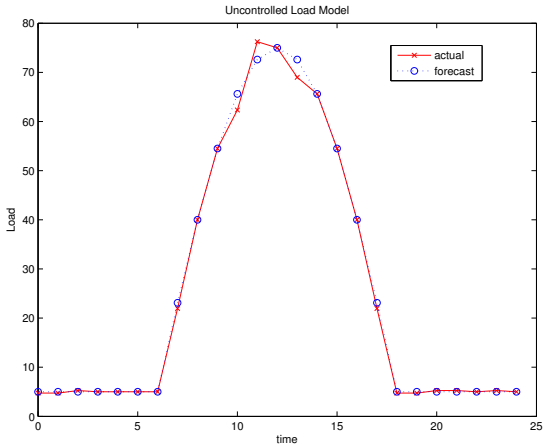

Fig. 6. The uncontrolled load is modeled as piece-wise sinusoidal. Here $start = 6, end = 18$ (6 a.m. to 6 p.m.) $low = 5$, and $high = 75$.

4) Utility Price: Similarly to the uncontrolled load model the utility pricing model is piecewise sinusoidal with multiplicative noise. Figure 7 depicts a realization of the utility pricing model through a 24 hour episode.

$$forecast = \begin{cases} low & t < start \\ \sin\left(\frac{(t-start)\pi}{end-start}\right)(high-low)+low & otherwise \\ low & t > end \end{cases}$$

$$actual = \begin{cases} forecast\,(1-\Delta_{UP}) & w.p.0.2 \\ forecast & w.p.0.6 \\ forecast\,(1+\Delta_{UP}) & w.p.0.2 \end{cases}$$

$$\Delta_{UP} = 0.05$$

5) Wind Speed/Power: The wind speed is modeled as a 1-step Markov process. Figure 8 depicts a 24 hour realization of the wind speed. A simplified linear model for wind power from wind speed is assumed. Wind speed is limited to be no less than 0 and no greater than 30.

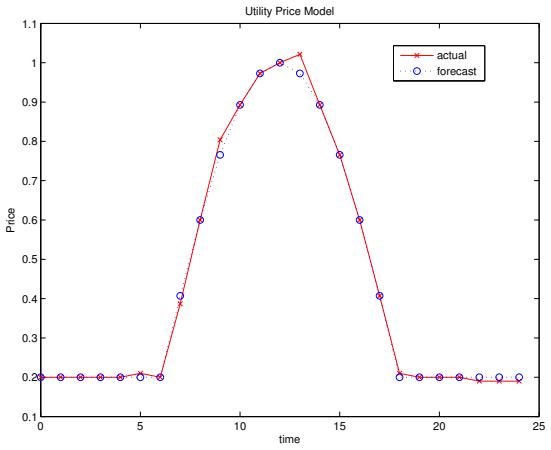

Fig. 7. The utility pricing model is piece-wise sinusoidal with multiplicative noise. Here $dayStart = 6$, $dayEnd = 18$, (6 a.m. to 6 p.m.), $low = 0.2$, and $high = 1$.

$$forecast = priorValue$$

$$actual = \begin{cases} forecast - \Delta_W & w.p.0.2 \\ forecast & w.p.0.6 \\ forecast + \Delta_W & w.p.0.2 \end{cases}$$

$$\Delta_W = 2$$

$$windPower = \alpha \times windSpeed$$

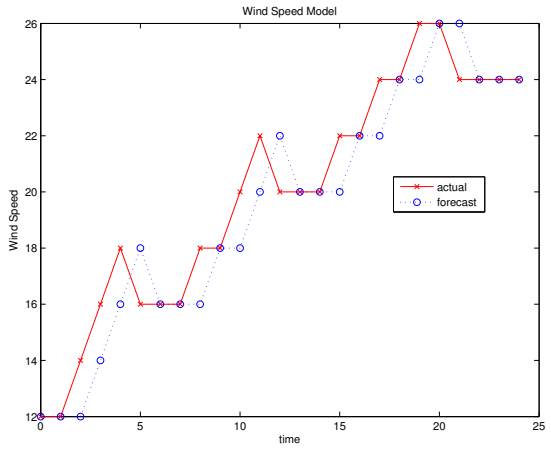

Fig. 8. Wind speed is modeled as a 1-step Markov process. Wind power is linearly proportional to wind speed.

6) Battery: The battery state is determined by user input up to capacity limits. The battery level is allowed to change by -25 (discharge), 0 (no action), or 25 (charge) within a given stage. The minimum capacity is 0 and the maximum capacity is 100.

7) Cost: The total cost function is given by eq. (5). The total cost is a weighted combination of the dollar and comfort costs given in eqs. (3) and (4) respectively. This weighting

will depend on user preference. Here the comfort point is the optimal indoor temperature determined by the user.

$$dollarCost = \sum_{i}^{N} netLoad_i \times gridPrice_i \quad (3)$$

$$netLoad_i = ACLoad_i + unctrlLoad_i + dBatt_i - windPower_i$$

$$comfortCost = \sum_{i}^{N} (comfortPoint - indoorTemp_i)^2 \quad (4)$$

$$totalCost = 0.6\,(dollarCost) + 0.4\,(comfortCost) \quad (5)$$

D. Stochastic Dynamic Programming Results

In order to implement the stochastic dynamic programming algorithm a finite horizon was assumed. The most natural episode was a 24 hour period starting at midnight and continuing until midnight of the following day with a stage decision every hour on the hour. This allowed optimal decision making for every state starting from the final decision stage (11p.m.), continuing backwards until the initial stage (12a.m.). Costs and state transitions following the models described in the previous section provided a means for calculating optimal decisions for every state at every stage using stochastic dynamic programming. Figure 9 shows the evolution of indoor and outdoor state variables with respect to the set-point and comfort point given optimal decision making.

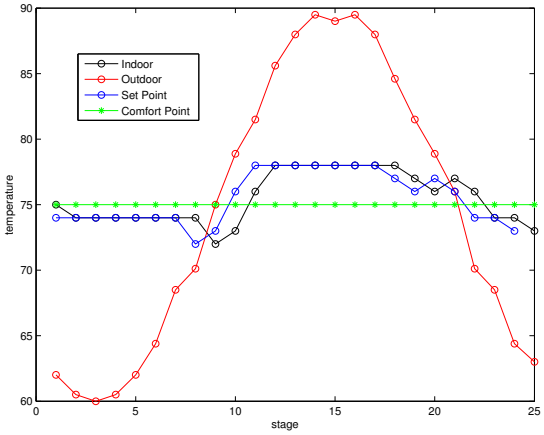

Fig. 9. Evolution of indoor and outdoor temperatures given optimal decision making dictated by stochastic dynamic programming

Figure 9 illustrates some of the interesting characteristics of optimal decision making in a stochastic environment. The algorithm chooses to reduce the set-point in the early morning hours to allow for some discomfort in balance of dollar cost. This effect is considerably more pronounced during peak hours when demand for load, utility price, and outdoor temperatures are high. Figure 10 considers the individual load components, and sheds some insight into the battery charging/discharging decision process. Battery charging occurs in the early morning hours while electricity is cheap and demand is low. The battery discharges during peak hours to meet high prices and demand for load. Figure 11 outlines the cost components associated with this example. Here comfort cost is maintained near zero while dollar costs vary in proportion to load demand and utility costs.

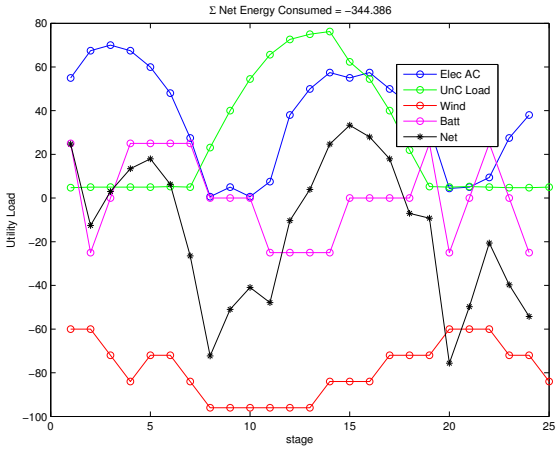

Fig. 10. Utility load profile for each modeled component. Note that in this instance the net energy consumed is negative due to the availability of wind power.

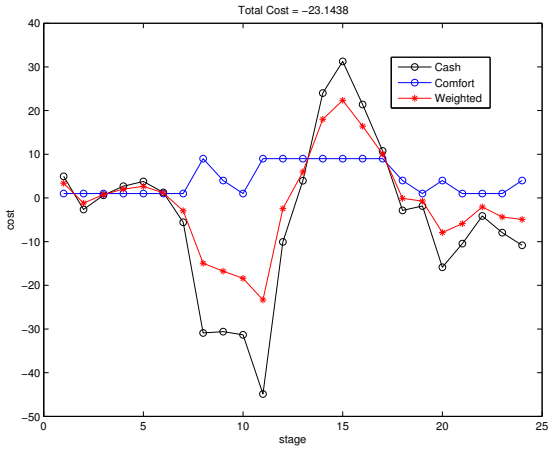

Fig. 11. Cost profile components. Comfort cost is associated with deviations in indoor temperature, dollar costs are associated with utility being used/supplied and the utility price. The weighted cost is the cost function used by the stochastic dynamic programming algorithm in determining optimal decisions.

E. Stochastic Dynamic Programming Discussion

While the stochastic dynamic programming solution found here seems to work well enough for the problem that was constructed, the solution was still very computationally

intensive and relied upon discretized states with known transition probabilities. In a real world environment with unknown state transition models and cost functions, optimal decision making in this fashion is unlikely. Additionally, state variables will not be discrete values from episode to episode, and a complete solution would require optimal decision making solutions for an infinite number of possible states. In the next section an adaptive algorithm capable of dealing with some of the real world issues addressed here is discussed.

III. A Reinforcement Learning Approach

A more generalized solution to optimizing the smart system decision making process would do without some of the assumptions made under the stochastic dynamic programming environment. Namely, it would be nice to make intelligent decisions from data and experience gathered without knowing much about the system state transition probability rules or costs, and without forecasting future state values. Without knowledge of these components there is no way to immediately make optimal decisions. In this type of environment optimal decisions must be learned from experience.

From a high level the smart system optimization problem is one of mapping decisions to costs, and choosing those decisions which minimize costs to be optimal. Viewing the problem in this way greatly simplifies the system architecture, and refocuses the optimization problem on learning which decisions, made from a particular state, minimize cost. Figure 12 outlines the basic system flow, where decisions are based on past costs.

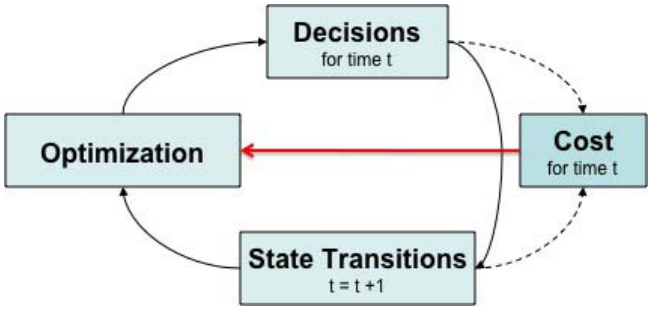

Fig. 12. In the simplified reinforcement learning optimization architecture decisions are based on past experience alone.

A. Adapted Softmax with Neighborhood Update Model

The softmax decision algorithm is based on choosing actions via a probability distribution [3]. Positive outcome (low cost) decisions increase the probability of choosing the same action in the future while negative outcome (high cost) decisions decrease the probability of choosing an action. Over time, the probability distributions should heavily favor optimal decisions. The softmax algorithm has been adapted here to reduce the number of probabilities that need to be calculated. Instead of learning roughly 21 probabilities associated with each decision at each state, we can parameterize the distributions. Using Gaussian distributions reduces the problem to that of determining an optimal mean and variance over the decision space for each state. When a state/action combination is found which reduces total cost, the mean of the decision probability distribution for that state is updated. After being randomly initialized the mean is updated according to a learning rate every time a decision provides reduced costs. The deviation of the distribution is then updated in proportion to the size of the mean update. If the new, optimal mean of the decision distribution was close to the old one, the variance may be reduced. If the mean update was large the variance is maintained or increased. This follows from an exploration-exploitation optimization scheme in which early exploration yields erratic results and late exploitation takes advantage of learned rules for cost minimization.

Given the update rules described above we may learn at most once per stage. With approximately 800,000 states that must be learned, this process will take a very long time. In the smart system optimization problem we expect the state/decision costs to be relatively smooth. For example, the cost of placing the temperature set-point at 72 vs. 73 is likely very similar. This implies that we can share decision and cost information amongst states. Under the softmax framework we can now assume that the probability distribution parameters for neighboring states should be similar. Thus, while updating the parameter information for the current state, we can update the neighbor states according to a reduced weighting coefficient. Similarly, neighbor best costs may be reduced given significant discrepancies. For example, best cost estimates are initialized to infinity. This means that the first time this state is encountered the probability distribution governing the decisions will be updated regardless of the cost result. This type of false update may be avoided by initializing unobserved states to values higher, but on the same order as neighbor states. Currently, neighbor costs are updated as 1.5x the base cost.

B. Adapted Softmax with Neighborhood Update Results

The method of adapted softmax with neighborhood updates was applied over 100 days (episodes) of continuous running with no prior knowledge of state transitions or cost structure. The softmax distributions for all states were initialized with randomly distributed means and deviations of 1 for the temperature set-point and 15 for the battery charge/discharge. The neighbor update weights were [0.7 0.3 0.2] (update two neighbors away), for immediately adjacent neighbors. The product of neighbor weights was used for non-adjacent neighbors.

Figure 13 shows the temperature results after running the algorithm for 100 days after random initialization. While the algorithm is not producing the sophisticated results observed in the stochastic dynamic programming example, it is able to maintain the temperature around the comfort point even though it has no prior knowledge of its existence. Figure

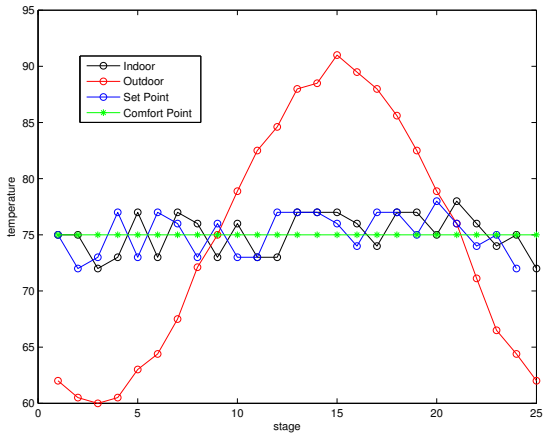

Fig. 13. Temperature profiles achieved given the softmax-neighbor update method. Notice how the indoor temperature hovers around the comfort point.

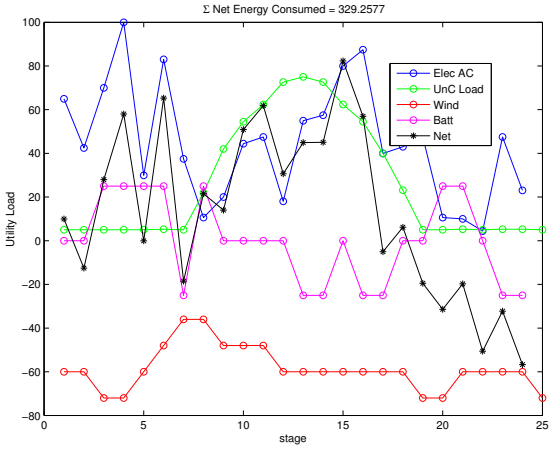

Fig. 14. Individual load components using the softmax-neighborhood update method. Notice that battery charging occurs in the early morning hours, and discharging occurs during peak pricing and loads.

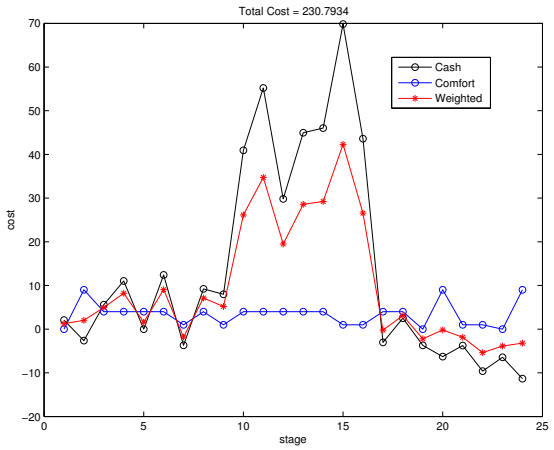

Fig. 15. Cost component example for the softmax neighbor update method.

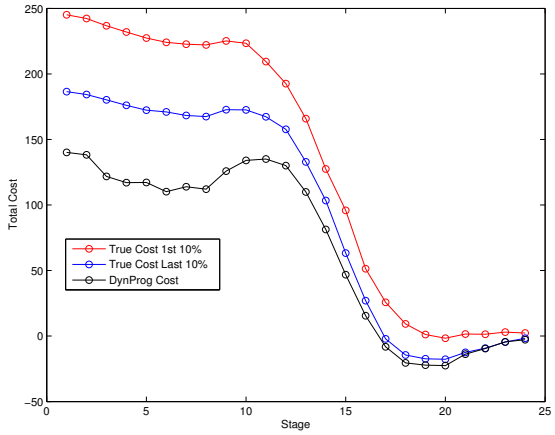

Fig. 16. Total cost by stage. Red depicts the average cost of the first 10 days, blue depicts the average cost of the final 10 days (91-100), and black is the optimal cost as determined by stochastic dynamic programming.

14 shows the individual load components for this example. While, here again, the decisions to charge/discharge the battery are not as sophisticated as those observed under the stochastic dynamic programming scheme, in general the battery charges in the early morning hours when costs are low, and discharges at peak hours when costs and demand are high. Figure 15 depicts the cost function for the example considered here.

Figure 16 displays the stage costs for the softmax-neighborhood update method with respect to the stochastic dynamic programming costs over a 100 day training period. During the first 10 days the softmax distributions are generally randomized, but significant improvement is observed by the final 10 days. While the stochastic dynamic programming solution does significantly better still, it should be noted that any type of softmax approach will suffer with respect to hard optimal decision making.

IV. Conclusion/Future Work

Traditional stochastic dynamic programming is difficult to implement for real smart systems given the lack of state variable models and infinite state space realizations. Even severe discretization of the state space results in over 14 million state/action combinations. By combining ideas from softmax decision making routines as well as neighbor updates an algorithm may be implemented which does not require state forecasts or state variable and cost models. Optimal decisions may be asymptotically learned for any given system. This avoids the out of the box tuning required for implementing a smart system in a new environment. The algorithm presented here is also able to operate in non-stationary environments in which the state variable models, transition rules, and cost functions are subject to change.

Future work should be focused on three areas, updating the transition probabilities to allow policy iteration methods, moving to a continuous framework by possibly implementing

competitive learning schemes, and investigating the ability of forecasting combined with non-stochastic dynamic programming to provide a solution set. The simplest move forward is to update the existing algorithm to track the transition probabilities explicitly. Developing a transition probability model will allow for the development of a policy iteration framework, allowing the algorithm to utilize additional knowledge about which decisions are causing increased cost scenarios within an episode. A move to a continuous state-space scheme may be implemented by considering a competitive learning framework in which states are discretized to a node on an existing mesh of possible states. Each node on the mesh will contain information about the costs and optimal decisions associated with each state, and may adapt as in a self organizing map routine, to better account for often encountered states as models are learned. Non-stochastic dynamic programming may be used when accurate values of the forecasted states are available. The method boils down to a simple linear programming routine which may be run quickly to evaluate many different forecasts, and associated optimal decisions. These solution sets may provide valuable insights into the cost/decision landscape for a given state and allow further generalization. Additionally, updates to the *Energy Box* model [1] which include load scheduling scenarios instead of only load shedding will provide additional insight into the smart system problem.

ACKNOWLEDGMENT

The authors would like to gratefully acknowledge the support of the Renewable Energy and Island Sustainability (R.E.I.S.) program, as well as support from the National Science Foundation under grant NSF-ECCS0938344.

REFERENCES

[1] Livengood and Larsen, *The Energy Box: Locally Automated Optimal Control of Residential Electricity Usage*, Service Science 1 (1) 2009

[2] R. Sutton, A. Barto, *Reinforcement Learning An Introduction*, MIT Press, Cambridge, MA, 1998

[3] J. S. Bridle, *Training stochastic model recognition algorithms as networks can lead to maximum mutual information estimates of parameters*, In D. S. Touretzky (ed.), Advances in Neural Information Processing Systems: Proceedings of the 1989 Conference, pp. 211-217, Morgan Kaufmann, San Mateo, CA, 1990

[4] Bertsekas, *Dynamic Programming and Optimal Control*, vol. 1, 3rd ed., Athena Scientific, Belmont, MA, 2005

[5] Constantopoulos, P., F. C. Schweppe, R.C. Larson, *ESTIA: A Real-Time Consumer Control Scheme for Space Conditioning Usage under Spot Electricity Pricing*, Computers and Operations Research 18(8) 751-765, 1991

[6] Ilic, M.; Black, J.W.; Watz, J.L , *Potential Benefits of Implementing Load Control*, Proceedings IEEE Power Engineering Society Winter Meeting, 2002 , Vol. 1 , pp. 177 -182, New York, NY, January 2002