Autonomous HVAC Control, A Reinforcement Learning Approach

Enda Barrett[1,2] and Stephen Linder[1,2](✉)

[1] Schneider Electric, Cityeast Business Park, Galway, Ireland
[2] Schneider Electric, 800 Federal Street, Andover, MA 01810-1067, USA
{Enda.Barrett,Stephen.Linder}@schneider-electric.com
http://www.schneider-electric.com/

Abstract. Recent high profile developments of autonomous learning thermostats by companies such as Nest Labs and Honeywell have brought to the fore the possibility of ever greater numbers of intelligent devices permeating our homes and working environments into the future. However, the specific learning approaches and methodologies utilised by these devices have never been made public. In fact little information is known as to the specifics of how these devices operate and learn about their environments or the users who use them. This paper proposes a suitable learning architecture for such an intelligent thermostat in the hope that it will benefit further investigation by the research community. Our architecture comprises a number of different learning methods each of which contributes to create a complete autonomous thermostat capable of controlling a HVAC system. A novel state action space formalism is proposed to enable a Reinforcement Learning agent to successfully control the HVAC system by optimising both occupant comfort and energy costs. Our results show that the learning thermostat can achieve cost savings of 10% over a programmable thermostat, whilst maintaining high occupant comfort standards.

Keywords: HVAC control · Reinforcement learning · Bayesian learning

1 Introduction

Thermostats for controlling Heating, Ventilation and Air Conditioning (HVAC) systems in the home and office can largely be broken into two main categories: programmable and manual. Programmable thermostats allow the user to schedule heating and cooling to achieve patterns that work best for one's schedule. A thermal set-point is specified by a user and it governs the temperature and humidity levels that must be reached when the controller is active. Manual thermostats are non-programmable and require an external operator (human) to turn on and off the functions of heating and cooling as required. Manual thermostats are usually cheaper than their programmable counterparts.

Recently there has been a surge in the development of intelligent thermostats which boost the ability to autonomously control HVAC systems. These include offers from companies such as Nest Labs[3] and Honeywell[2]. They often only require the user to enter temperature set-points, while the schedule is learned automatically, with the objective of minimizing energy consumption while still allowing for occupant comfort. The unit attempts to learn a user's preference over time based on their manual adjustments and produce a schedule which is deemed optimal for the observed patterns of occupancy. The principal characteristics of these units is that they promote some notion of self-learning and automation; however, the user can usually override the learned schedule with a pre-fixed one.

In order to learn and effectively make decisions these systems rely on observations from sensory inputs about their environment. Commonly they use: temperature/humidity and motion sensors; an internal clock/calendar to track date/time; and external data sources such as the local weather conditions. Over time the goal of the learning thermostat is to refine its knowledge, update its understanding of the environment and make optimal decisions in accordance with its defined objective functions of maintaining occupant comfort and minimising cost.

To date, the specific learning approaches employed by companies such as Nest and Honeywell have never been publicly released and are guarded as trade secrets. In addition, there has been little activity from the research community to devise suitable open architectures for solving such problems. Therefore, this paper proposes a suitable learning architecture which utilises a number of learning methods capable of learning an optimal or near optimal control policy over time. The solution comprises a *Bayesian Learning* approach to accurately predict room occupancy over time and a *Reinforcement Learning* (Q-learning) method to learn a control policy for the thermostat unit itself. The reinforcement learning agent samples the output from the room occupancy prediction module to enable a better control solution.

In summary the principal contributions of this work are

– A learning architecture which can support occupancy prediction and HVAC control, concurrently optimising both user comfort and energy costs
– A novel state action space formalism for the individual learning approach which enables a multi-criteria optimisation solution.

The rest of this paper is structured as follows: *Background Research* provides an overview of relevant and related work in this field, including work specific to the learning approaches used and other applied learning work. *Markov Decision Processes & Learning Methods* describes the concepts and learning approaches used in this work. *HVAC Control* details specifics relating to how to apply these methods to the real world problem. *Initial Results* details our preliminary findings, leading finally to *Conclusions & Future Work.*

2 Background Research

A standard HVAC system can be considered to comprise two principal components, a heating/cooling element and a fan for circulating the air. In order to operate the system, the user generally specifies a setpoint value on the thermostat interface denoting the room conditions they require. Using the output from a temperature sensor the thermostat monitors the changing room conditions as a result of turning on the HVAC. Once the room temperature has reached the target setpoint the system is switched off. In more optimised systems the fan speeds can be adjusted to enable further optimisations over this simple scenario.

To autonomously control HVAC systems, a number of methods are required to enable the features depicted on modern controllers such as the Nest. Firstly a learning method must learn when to turn on and off heating and cooling, we refer to this as the thermostat control policy for which we employ a reinforcement learning method known as Q-learning. The goal of the learning process is to control the HVAC system i.e. turn it on and off, to ensure that the user's setpoint is maintained at the lowest possible cost. In addition as an aid to this control policy, Bayesian inference is utilised to predict room occupancy allowing for greater cost savings where unoccupied rooms need no longer be heated. An accurate prediction method is necessary in order to preemptively heat and cool rooms prior to being occupied. Occupancy sensors can only tell you when they detect whether or not a room is occupied, not when it is going to be occupied, thus only heating and cooling based on occupancy detection will likely result in a low comfort rating from users who will have to wait until the room reaches the set point temperature. The combination of these techniques provides an overall architecture capable of providing a solution to the problem. One of the key value propositions is the ability of each component to build up knowledge whilst operating directly with the environment, without prior experience.

Whilst there has been substantial activity in the commercial space with numerous patents filed in this area for both automating the control of HVAC systems entirely or partially through varying components of these systems, there has been little activity in the research community. To the best of the authors' knowledge this paper is the first application of Reinforcement Learning to this problem domain.

In the 1990s manufacturers such as Mitsubishi [15] developed advanced fuzzy rule bases for controlling air conditioning systems in buildings which greatly outperformed the ubiquitous bang-bang controllers. The fuzzy rule systems allowed for intermediary control states where the air conditioning system could alternate between different fan speeds, humidity and temperature based on the environmental observations to reduce energy consumption and improve occupant comfort. Fuzzy systems rely on user defined rules which are collectively termed the fuzzy rule base. The output of the rules are combined to produce a smooth control response which creates smaller deviations around the temperature set points. Patents [9] [1] [4] describe a variety of fuzzy logic control methods ranging from the determination of thermal set-points in a HVAC system to methods for controlling HVAC to maximise occupant comfort in the automotive sector.

More recently, a patent filed by Nest [13] on 19 October 2012 describes a thermostat which uses machine learning and offers a taxonomy of learning approaches over which its claims are held. However the disclosure does not present a description of how these methods are applied or even which methods are used in their implementation. On the Web there have been a number of hardware teardowns of the Nest thermostat, but to date little is known about the software controling the device.

In a user trial on a number of homes in the US and UK by Scott et. al. [18] (entitled PreHeat) RFID tags were added to the house keys of each occupant. Their home heating solution used occupancy sensing and prediction to better estimate when to heat the homes. Their results demonstrated substantial savings over a pre-scheduled heating solution for heating and cooling rooms in a house.

In a more general context, approaches from learning theory have been successfully applied to automated control problems across a range of domains. Dutreilh et. al. [12] devised a Q-learning approach for allocating resources to applications in the cloud. Gerald Tesauro created TD-Gammon [21], a reinforcement learning artificially intelligent agent capable of playing backgammon to international level. Other notable successes include workflow scheduling [6], traffic light control [24] and application scaling [7] in computational clouds. The important novelty common to these works is not so much their extension to learning theory but more so their application of learning theory to solve a real world problem.

3 Markov Decision Processes and Learning Methods

3.1 Markov Decision Processes

Markov Decision Processes (MDPs) are a particular mathematical framework suited to modelling decision making under uncertainty. A MDP can typically be represented as a four tuple consisting of states, actions, transition probabilities and rewards.

- S, represents the environmental state space;
- A, represents the total action space;
- $p(.|s,a)$, defines a probability distribution governing state transitions $s_{t+1} \sim p(.|s_t, a_t)$;
- $q(.|s,a)$, defines a probability distribution governing the rewards received $R(s_t, a_t) \sim q(.|s_t, a_t)$;

S is the set of all possible states represents the agent's observable world. The agent learning experience can be broken up into discrete time periods. At the end of each time period t the agent occupies state $s_t \in S$. The agent chooses an action $a_t \in A(s_t)$, where $A(s_t)$ is the set of all possible actions within state s_t. The execution of the chosen action, results in a state transition to s_{t+1} and an immediate numerical reward $R(s_t, a_t)$. The state transition probability $p(s_{t+1}|s_t, a_t)$ governs the likelihood that the agent will transition to state s_{t+1} as a result of choosing a_t in s_t. The numerical reward received upon arrival at the next state is governed by $q(s_{t+1}|s_t, a_t)$ and is indicative as to the benefit of choosing a_t whilst in s_t.

The solution of a MDP results in the output of a policy π, denoting a mapping from states to actions, guiding the agent's decisions over the entire learning period.

In the specific case where a complete environmental model is known, i.e. (S, A, p, q) are fully observable, the problem reduces to a planning problem [16] and can be solved using traditional dynamic programming techniques such as value iteration. However if there is no complete model available, which is often common with real world problems, then one must either attempt to approximate the missing model (Model Based Reinforcement Learning) or directly estimate the value function or policy (Model Free Reinforcement Learning). Model based methods use statistical techniques in order to approximate the missing model [14], whereas model free learners attempt to directly approximate a control policy through environmental interactions.

In this work we choose to utilise a model free learning method known as Q-learning. The approach has been widely applied to real-world problems, which allows for stricter comparisons with previous work and is capable of finding an optimal or near optimal control policy in a reasonable time.

3.2 Reinforcement Learning

Modeling the HVAC control problem as a MDP enables us to design a solution which can effectively handle environmental uncertainty. However as with most real world learning problems, we have no prior knowledge of the complete environmental model, the distribution of rewards or transition probabilities. Therefore, solutions from Dynamic Programming such as Value Iteration or Policy Iteration cannot be used to generate an optimal policy π for these problems. As an alternative to Dynamic Programming, model free Reinforcement Learning methods such as Q-learning [23] can be used to generate optimal policies in the absence of a complete environmental model.

Q-learning belongs to a collection of algorithms called Temporal Difference (TD) methods. Not requiring a complete model of the environment, TD methods possess a significant advantage and have the capability of being able to make predictions incrementally and in an online fashion. We choose to use Q-learning for this research, not for its demonstrated efficacy within the domain but more for its wide applicability to applied domains published previously [10] [8] [22]. The update rule for Q-learning is defined as

$$Q(s,a) \leftarrow Q(s,a) + \alpha[r + \gamma Q(s',a') - Q(s,a)] \tag{1}$$

and calculated each time a state is reached which is nonterminal. Approximations of $Q^{\pi}(s,a)$ which are indicative as to the benefit of taking action a while in state s, are calculated after each time interval. Actions are chosen based on π, the policy being followed. A number of action selection policies can be used to decide what action to select whilst occupying a particular state, examples include ϵ-greedy, softmax and unbiased sampling [20]. The goal of these selection strategies is often to carefully balance exploration and exploitation to yield the best possible results in the shortest possible time frame. Over time the actions selected should converge to the optimal where the agents consistently choose actions which present it with the greatest amount of cumulative reward over the course of the interaction. In the case of ϵ-greedy, the goal is to choose the best action most of the time except for a certain amount of time governed by ϵ when the agent chooses an exploratory action. Let $A'(s) \subseteq A(s)$, be the set of all non-greedy actions. The probability of selection for each non-greedy action is reduced to $\frac{\epsilon}{|A'(s)|}$, resulting in a probability of $1 - \epsilon$ for the greedy strategy.

Estimated action values for each state action pair $Q^{\pi}(s, a)$ can be represented in tabular form or as part of a generalised function approximator. The goal of the learning agent is to maximize its returns in the long run, often forgoing short term gains in place of long term benefits. By introducing a discount factor γ, $(0 < \gamma < 1)$, an agent's degree of myopia can be controlled. A value close to 1 for γ assigns a greater weight to future rewards, while a value close to 0 considers only the most recent rewards. Reinforcement learning based approaches are capable of reasoning over multiple actions, choosing only those which yield the greatest cumulative reward over the entire duration of the episode. The steps involved in Q-learning are depicted by Algorithm 1.

Algorithm 1. Reinforcement Learning Algorithm (Q-learning)

```
Initialize $Q(s, a)$ arbitrarily
  Repeat (for each episode)
  Initialize $s$
  repeat
    Choose $a$ from $s$ using policy derived from Q ($\epsilon$-greedy)
    Take action $a$ and observe $r$, $s'$
    $Q(s, a) \leftarrow Q(s, a) + \alpha[r + \gamma \max_{a'} Q(s', a') - Q(s, a)]$
    $s \leftarrow s'$;
  until $s$ is terminal
```

Q-learning can often require significant experience within a given environment in order to learn a good policy. This is largely determined by the size of the state and action space. In particular, tabular Q-learning methods require continuous updating of the value estimates through repeatedly revisiting the states and choosing actions in the environment. As the size of the state action space grows, this problem can become more pronounced (often referred to as the curse of dimensionality), where each additional state or action variable added, increases the problem size exponentially. For the purposes of this work we utilise tabular Q-learning methods but convergence times could be improved by utilising techniques such as parallel learning[5] or function approximation.

3.3 Bayesian Inference

The final part of the problem requires a solution for occupancy prediction in order to make better judgements as to when one is required to heat and cool the space under control.

For this work we employ a Bayesian inference technique in order to make predictions. Bayes theorem is a mathematical framework which allows for the integration of one's observations into one's beliefs. The posterior probability $P'(X = x|e)$, denoting the probability that a random variable X has a value equal to x given experience e can be computed via

$$P(Y|X) = \frac{P(X|Y)P(Y)}{P(X)} \tag{2}$$

which requires one conditional probability $P(X|Y)$ and two unconditional probabilities $(P(Y), P(X))$ to compute a single conditional posterior probability $P(Y|X)$ [17].

Bayesian learning algorithms generally combine Bayesian inference (Bayes rule) and agent learning to build up probabilistic knowledge about a given domain. Statistical inference methods can prove particularly useful when attempting to approximate the likelihood of an event occurring given past experiences. By providing an estimated occupancy model our overall solution is capable of reducing the energy consumption by only heating and cooling when necessary.

4 HVAC Control

This section discusses the specifics of applying each technology to the domain. We present a novel state action space formalism for Q-learning which enables it to effectively control heating and cooling in an online manner. In addition we describe a method to predict occupancy using a modified Bayes rule and corresponding update function.

4.1 Occupancy Prediction

The specific inference rule applied for occupancy prediction was originally defined by David Spiegelhalter [19] and further extended by Prashant Doshi [11]. It employs a modified Bayes rule, where all that is required to compute the posterior probability is an initial prior probability and subsequent environmental experience. The approach involves maintaining an experience counter $Expc$ for each observation and updating the distribution according to equations (3) and (4). These equations define the update rules for approximating the likelihood of occupancy based on past experience [1]

$$P'(s = s'|a, s = s) = \frac{P(s = s'|a, s = s) \times Expc + 1}{Expc'} \tag{3}$$

where equation 4 ensures that the probability distribution over the total number of possibilities sums up to 1. y represents the set of all possible next states achievable from s minus s' the actual next state resulting from action selection a.

[1] $Expc'$ is the incremented counter, $Expc' = Expc + 1$.

$$P'(s = y|a, s = s) = \frac{P(s = y|a, s = s) \times Expc}{Expc'} \tag{4}$$

From an implementation perspective the approach requires an occupancy sensor to provide it with the necessary evaluative feedback in order to update the model over time. Every minute the learning agent queries the sensor which returns a boolean result (true or false) depending on whether or not the room was occupied at that time. Based on the response a binomial distribution is updated accordingly using equations (3) and (4). This simple solution is surprisingly efficient at making predictions and doesn't require large amounts of environmental experience.

4.2 HVAC Control Using Q-learning

The HVAC system employs Q-learning by framing the environment as a MDP. In order to accurately solve the problem we must first define the set of states S and actions A i.e. the agent's observable world and the actions it can take in it:

– rt : is the room temperature (*source*: temperature sensor, *unit*: °C);
– tto : is the time to occupancy (*source*: occupancy predictor, *unit*: minutes);
– ot : is the outside temperature (*source*: weather station, *unit*: °C);

The second thing we define is the action space A which consists of the following four choices:

– $Heat_{on}$: turns on heating;
– $Heat_{off}$: turns off heating;
– $Cool_{on}$: turns on cooling;
– $Cool_{off}$: turns off cooling;

The idea is to try to keep the number of states and actions low so that the problem remains within the bounds of tractability. However even though there are only three state variables, each state variable can take on a wide range of values quickly creating a relatively large state space. For instance the indoor temperature could range from the low teens to the mid to high twenties (12°C to 27°C). The outside temperature could vary from region to region, but in places such as North America it would not be uncommon to experience highs of 40°C in the Summer and lows of −20°C in the depths of Winter. In addition the time to occupancy tto (minutes) at any particular moment may be a number of hours away, substantially increasing the size of the state space.

HVAC controller actions are executed at discrete time intervals known as epoches. For instance an epoch of 5 minutes assumes that a controlling action for the HVAC system may be executed at either 10:00 or 10:05, but not at 10:02. The granularity is a configurable parameter and can be adjusted to ensure an optimal configuration such as at minutely intervals. At the end of each epoch the learning agent observes the current state of the environment and chooses whether or not to execute an automated HVAC action (turn on or off).

The transition probabilities T i.e. the likelihood of transitioning between states after executing particular actions is not known apriori so this problem cannot be solved using Dynamic Programming methods such as Policy or Value iteration. In addition we do not attempt to estimate T, instead Q-learning observes the consequences of T and adjusts accordingly.

The rewards achievable by the learning agent are distributed in accordance with certain scenarios that arise and are scalar in value. A setpoint variable *sp* specifies the user defined objective temperature setting. Rewards are calculated as follows:

1. (Room.occupied = $false$) & (Action = $Heat_{on}$) & (rt > sp || rt < sp); $R = -1$
2. (Room.occupied = $true$) & (Action = $Heat_{on}$) & (rt > sp || rt < sp); $R = -3$
3. (Room.occupied = $false$) & (Action = $Heat_{on}$) & (rt = sp); $R = -1$
4. (Room.occupied = $true$) & (Action = $Heat_{on}$) & (rt = sp); $R = -1$
5. (Room.occupied = $false$) & (Action = $Heat_{off}$) & (rt < sp || sp > rt); $R = 0$
6. (Action = $Heat_{off}$) & (rt = sp); $R = 0$
7. (Room.occupied = $true$) & (Action = $Heat_{off}$) & (rt < sp || sp > rt); $R = -3$

We assume a threshold around the setpoint of plus or minus 1°C. So if the user specifies a setpoint temperature of 23°C, the variable *sp* will range from $22°C - 24°C$. Each scenario listed above determines the rewards achievable as a result of choosing an action a within a particular state s. We haven't included cooling as part of the scenario, but the same rules will govern its action selection also. *Scenarios 1,3,4* result in a reward of -1. This cost is representative of the cost that would be incurred for operating the heating control of the HVAC unit per time step. This could easily be extended to include real time energy pricing costs if necessary. For *Scenarios 2 and 7* a fixed penalty is applied resulting in a reward of -3. The penalty chosen does not need to be specifically -3 but it must be greater than the unit cost of the HVAC operation i.e. (-1). For *Scenarios 2 and 7*, irregardless of the action chosen i.e. $Heat_{on}, Heat_{off}$ the penalty is applied because the setpoint temperature specified by the user has not been met and the room is presently occupied. *Scenarios 5 and 6* result in a reward of 0, i.e. no cost is incurred as the heating is turned off and either the room is not occupied *(Scenario 5)* or the setpoint has already been met *(Scenario 6)*.

5 Initial Results

This section describes our initial results with the autonomous thermostat controller. For the purposes of this research we conducted evaluations via simulation only. We present results for both occupancy prediction and thermostat control, demonstrating empirically the efficacy of the solutions as possible approaches for solving the problem.

5.1 Occupancy Prediction

We evaluated our occupancy prediction method by creating an occupancy model, which simulates when the room is occupied. We assume an occupancy sensor is always available, however we vary the accuracy of this sensor using a Gaussian distribution of mean zero and standard deviation one. For a Gaussian or normal distribution, 70% of the time a random variable X takes on a value x which will fall within one standard deviation either side of the mean. 95% of the time the value will fall within two standard deviations and 99% of the time it will fall within three standard deviations of the mean. Thus we can vary the accuracy of the sensor by introducing statistical noise and returning either a false positive or negative depending on the actual result. This is to replicate scenarios which may cause false positives such as a cat/dog moving or false negatives such as if a person is in the sensor blind spot. We argue that a prediction approach should be capable of handling such sensing errors which would arise under normal operating conditions.

The goal of the Bayesian learner is to approximate the user's patterns of occupancy as closely as possible based solely on the learners observations. To do this, the learner continuously updates its beliefs, represented probabilistically, over time. In order to evaluate the approach we employ the Kullback-Liebler (KL) divergence to determine the difference between the binomial distribution of the learner and the true values as learning progresses. The KL divergence, sometimes referred to as information gain or relative entropy gives a measure of the distance between two probability distributions. For two probability distributions P and Q, the KL divergence is

$$D_{KL}(P \parallel Q) = \sum_i ln \frac{P(i)}{Q(i)} P(i) \tag{5}$$

Note that the KL divergence is not a true metric and is not symmetrical, meaning that the KL divergence from P to Q is not equal to the KL divergence from Q to P. If P and Q are identical then $D_{KL} = 0$.

Figure 1 displays how predictions progress over time under three separate settings. The average KL divergence between the learned occupancy model and the true model is plotted for each setting. Note that the true model is not strictly ground truth occupancy, as it incorporates the variance in sensing for a more realistic comparison. It's worth noting that in a given week the learner will only sample each time period once, this means that over the course of a month the learner will have four bouts of experience to train its model. By grouping days into categories such as "working week" or "weekends", the learning time for occupancy prediction could be dramatically reduced as the information learned over a number of days could be aggregated. However we choose to treat each day as an independent event in order to present a clearer evaluation of the time it takes to learn an occupancy model. Figure 1 plots the course of the model learning process over 150 days. The first two curves detail the affect of the sensing error on the learning process, whilst the third demonstrates sensing error and model shift. Model

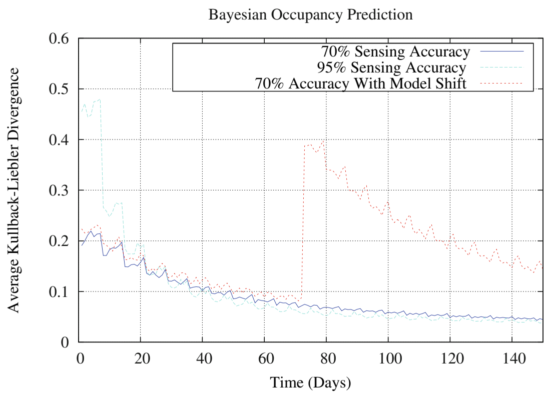

Fig. 1. Occupancy prediction evaluation with varying sensing error and model change

shift is intended to represent a scenario where the underlying occupancy pattern changes entirely, i.e. a user suddenly begins to work nights instead of days and their pattern of occupancy changes to reflect this.

When the sensor is only 70% accurate, the initial difference between the two distributions is less than when the sensor is 95% accurate. The reason for this is because the initial equiprobable binomial distribution is not too dissimilar to the true model as the true model has greater variance due to the sensing errors. For this reason when the accuracy of the sensor is switched to 95% the initial divergence is much higher as distributions differ by greater amounts. However, with the greater accuracy in sensing the learner is capable of better approximations of the true model, improving upon the 70% curve after approximately 40 days. It's clear that in both cases only a small amount of experience is required to make a good approximation of the underlying model. After 20 days the learner has built up a good predictive model of occupancy in both cases.

The last curve plots the effect of model shift on the approach. How a learning approach recovers from a shift in its underlying model is an important feature of online learning methods. Offline methods usually have to be retrained once such an event occurs but online methods should show adaptability to this type of behaviour. Model shifts are always challenging from a learning perspective because the agent has already significant past experience pointing to something which is no longer valid and how it adjusts its estimates determines its efficacy in the domain. If the agent simply disregards all the previous estimates in favour of the most recent, a temporary change could easily skew the predictive power of your solution.

After 70 days the occupancy behaviour of the user changes causing a jump in divergence. The key thing here is to note the recovery, i.e. within the space of 50 days the learner has returned to making good approximations of this new underlying model. The approach generally demonstrates good approximations without any prior knowledge, however the accuracy of the solution will always be constrained by the quality of the sensing devices, through which predictions are attained.

The output of occupancy prediction is a multinomial distribution governing the probability of occupancy for specific times over the course of a given day. This distribution is utilised by the RL agent (Q-learning) to control heating and cooling where the distribution is sampled in order to approximate a value for "time to occupancy".

5.2 Autonomous Thermostat Control via Q-learning

Simulation Environment. In order to ensure the repeatability of our experiments we simulate the heating of a room by defining a heat transfer rate, an input heating rate and calculating the temperature changes for each time step. Equation 6 describes the calculation of the heat transfer rate in Watts,

$$Heat_{transfer} = uValue \times surfaceArea \times (rt - ot) \tag{6}$$

where $rt - ot$ is the difference in temperature between the internal temperature and the external outside temperature. The u-value[2] is given in units of W/m^2K. By dividing the thickness in (m) of the materials (plaster, slab, screed, etc) by their manufacturer stated resistivity values one can compute an approximate u-value for the building/room. It is generally given as $1/totalResistance$. By measuring the total surface area of the room (m^2) one can work out the heat transfer rate i.e. the amount of heat energy in Watts leaving the room at any given moment.

To model the effects with respect to temperature changes we simulate using the following configuration. The specific heat of air is the amount of energy (Joules) required to raise the temperature of $1Kg$ of air by $1°K$ and works out to be approximately $718J/KgK$ given atmospheric pressure of 1 atm and air density of $1.3Kg/m^3$. For simulation purposes we assume a resistive heater is heating the room and it's 100% efficient, meaning that if it's rated 1kW it is outputting 1kW of heat energy into the room.

We modelled the effects of heating on a perfectly uniform cubed shaped room which has a surface area of $54m^2$. We assume the ceiling, walls and floor are insulated with each having u-values of 0.4, 0.6, 0.5 respectively. If the outside temperature ot at time t is $10°C$ and the inside room temperature rt is $20°C$, then the temperature difference between inside and out is $10°C$. Using equation 6 one can compute the heat transfer for each component i.e. the heat escaping through the ceiling would be given by $0.4 \times 9 \times 10 = 36W$. Obviously rooms are often not entirely uniform but for simulation purposes it's a reasonable assumption. By aggregating the heat transfer of each component (ceiling, walls, floor) at time t we can determine the total heat transfer in Watts. We then subtract this value from the heat input to determine the net heat gain into the room. Say the simulated room has a heat transfer rate of $300W$, then 1 minute of

[2] Can also be known as the r-value in some countries.

heating by a $1kW$ heater into this room would result in a temperature increase of $(700 \times 60)/718/(1.3 \times 27) = 1.67°C$, where $27m^3$ is the room volume. This approach allows us to model the temperature changes in a repeatable and reproducible manner. Whilst we do not observe all room parameters such as the heat generated by individuals occupying the room or windows/doors being left open, the state space is sufficiently informative enough to ensure a good measure of control is possible.

In order to get a measure of the outside temperature ot for our simulations, we utilised data supplied from the weather station situated at the National University of Ireland, Galway. The University provided us with five months of environmental data dating from 1 January 2013 to 31 May 2013. The data was sampled every minute and consists of temperature, humidity, wind speed and atmospheric pressure. For experimental purposes we focus solely on room heating as the temperatures within Ireland are relatively moderate and cooling systems are generally not required in many environments such as the home. However from both a learning and control viewpoint the same principles will still apply.

The goal of this research is to produce a control solution which can effectively combine both occupant comfort with energy cost savings. For comparative purposes we focus on comparing the costs for the "Always On" and "Programmable Control" methods, ignoring the "Manual Control" method as it's not a realistic comparison with our proposed solution as from a cost perspective it cannot be optimised any further. From a comfort analysis we focus on comparing the affect of different learning rates on Q-learning and show how occupant comfort can be improved by adjusting the configuration settings on the learning approach.

Online Q-learning vs HVAC "Always On". Figure 2 plots a comparison between an online Q-learning approach and an "Always on" solution. Online learning with respect to Q-learning means that the agent is arbitrarily initialised in the beginning and has no prior knowledge of the domain. The "Always on" method means that the HVAC system operates 24/7. Many users operate their HVAC systems in this way, as they often cannot understand how to program their thermostat properly or if they are sick/elderly. Figure 2 plots the monetary costs of both solutions over the period from 1 January 2013 to 31 May 2013. From a cost perspective, it's clear that the online Q-learning method combined with occupancy prediction is capable of operating the heating of the room at more than half of what the "Always on" solution costs. The total costs for heating the room for the period under consideration were €152.55 vs €344.15. The results demonstrate the significant savings that are achievable using adaptive control via Q-learning and occupancy prediction when compared to "Always on".

However one of the significant advantages of constantly running your HVAC system is that you can always be sure of the comfort of the environment where the set point temperature is always maintained. The goal of the learning solution proposed by this paper is achieve significant cost reduction whilst concurrently optimising the comfort levels of the end user, so we need to make sure that this occurs.

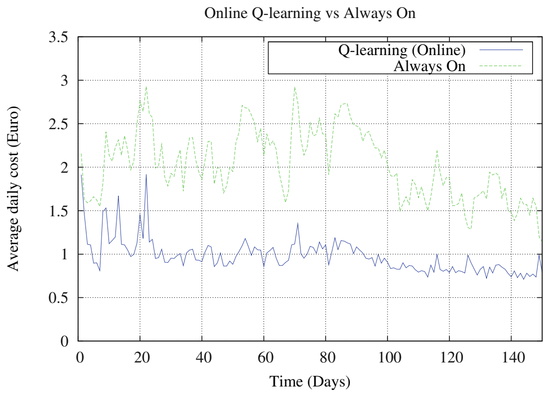

Fig. 2. Online Q-learning vs "Always on" control

Offline Q-learning Comfort Analysis. Offline learning involves an agent learning a good initial policy through simulation (offline) which can then be used when operating in the real world (online). It is commonly used to improve results over solely learning online. Since we interact with a simulator for our results we can utilise this method to demonstrate occupant comfort however online methods will still work, just more slowly.

Figure 3 plots the average amount of time in minutes when the temperature conditions were outside the setpoint temperature of $23°C$ with a threshold setting of plus or minus $1°C$. The results are carried out over multiple consecutive learning trials where the agent has no knowledge in the beginning, but carries forward its knowledge between trials. The graph considers two separate learning rates α with values of 0.1 and 0.5. The learning rate determines the amount by which the reinforcement learning agent backs up its value function estimates considering the new information presented to it. The higher the learning rate the shorter the amount of time it takes to learn a good policy, however too high a learning rate can lead to suboptimal policies where the approach takes too big of a step to correct the observed error in the estimate.

It's clear that after only a short number of learning trials the amount of time the comfort settings are not optimal has reduced to less than 40 minutes over the course of an entire day. Our results show that of these 40 minutes, 83% of these occur when the temperature is within $1°C$ of the threshold parameter. This means that whilst the environment is not optimal, the occupants would only experience mild discomfort. As the number of learning trials progresses this time reduces further. If the policy eventually turns to a completely greedy strategy over time then this should in theory drop to 0.

Offline Q-learning vs HVAC "Programmable Control". Continuing on from the previous section next we analyse the benefits of offline learning via simulation compared to a programmed schedule for operating the HVAC system. A

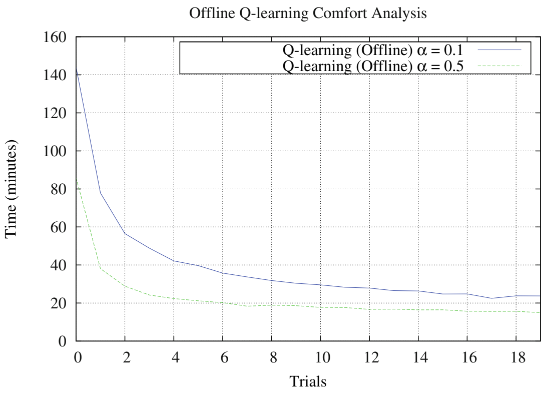

Fig. 3. Length of time in minutes when the setpoint temperature is not achieved but the room is occupied

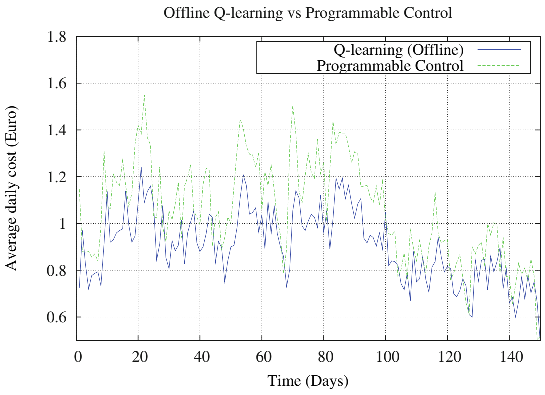

Fig. 4. Q-learning vs "Programmable Control"

number of other approaches could be utilised instead of offline training, i.e. function approximation would allow for generalisation over states and actions not yet visited over ones that have been if one were not able to perform offline training. In addition, parallel learning methods have also been proposed to achieve same where multiple independent thermostats could communicate in parallel in order to learn good policies. If one can simulate the environment, offline learning is a common technique where one can avoid the initial poor performance by yielding a good initial policy.

Figure 4 details the performance of offline learning against the programmed schedule. The schedule was designed by the facilities manager in the Schneider Electric Galway offices in accordance with how the building is currently operated. In the building, the HVAC systems are turned on at $7AM$ in the morning and go off at $8PM$ that evening. We simulated the occupancy so that on average, people begin at $8:30AM$ and finish at $6:30PM$. Figure 4 shows that the performance of the learning solution from a cost perspective out performs the programmed schedule with only two learning trials, i.e. it was trained offline for a single run and then applied to the problem. Overall there was a 10% improvement in costs as a result of employing learning over programmable schedules. Given enough learning experience figure 3 shows that the optimal setpoint temperatures can be achieved also proving that a combination of cost savings and occupant comfort can be achieved through this approach.

6 Conclusions and Future Work

This paper has demonstrated a reinforcement learning method combined with occupancy prediction capable of optimising the heating and cooling of a space autonomously with no prior information. Due to the limitations of our data set, our results focussed on heating only and demonstrated cost savings against two common strategies for controlling HVAC. In addition through offline learning via a simulator we demonstrated improved comfort and cost savings for the approaches in question.

In summary, if one carefully programs a thermostat and one's occupancy pattern is pretty regular, it's questionable how much energy savings can be achieved by a device such as a learning thermostat. The strategy is already optimal from a cost perspective. Thus we compared the approach against an "Always on" control method and "Programmable Control" method demonstrating cost reductions of 55% and 10% respectively in our simulated environments.

For future work the proposed state action space formalism could be extended further to give greater observation over the environment. In addition, methods from supervised learning such linear function approximation could be applied to generalise over the states and actions not yet visited based on those that have, reducing the time it takes to converge an optimal policy.

References

1. Fuzzy logic inference temp. controller for air conditioner, September 1, 1999. https://www.google.fr/patents/CN2336254Y?cl=en. cN Patent 2,336,254
2. Honeywell evohome, January 01, 2015. http://evohome.honeywell.com/
3. Nest thermostat, January 01, 2015. https://nest.com/thermostat/life-with-nest-thermostat
4. Ahmed, O.: Method and apparatus for determining a thermal setpoint in a hvac system, November 9, 2004. https://www.google.fr/patents/CA2289237C?cl=en. cA Patent 2,289,237
5. Barrett, E., Duggan, J., Howley, E.: A parallel framework for bayesian reinforcement learning. Connection Science **26**(1), 7–23 (2014)
6. Barrett, E., Howley, E., Duggan, J.: A learning architecture for scheduling workflow applications in the cloud. In: 2011 Ninth IEEE European Conference on Web Services (ECOWS), pp. 83–90. IEEE (2011)

7. Barrett, E., Howley, E., Duggan, J.: Applying reinforcement learning towards automating resource allocation and application scalability in the cloud. Concurrency and Computation: Practice and Experience (2012)
8. Choi, S., Yeung, D.Y.: Predictive q-routing: a memory-based reinforcement learning approach to adaptive tra c control. In: Advances in Neural Information Processing Systems 8, pp. 945–951 (1996)
9. Dage, G., Davis, L., Matteson, R., Sieja, T.: Method and system for controlling an automotive hvac system, July 22, 1998. https://www.google.fr/patents/EP0706682B1?cl=en. eP Patent 0,706,682
10. Dorigo, M., Gambardella, L.: Ant-q: a reinforcement learning approach to the traveling salesman problem. In: Proceedings of ML-95, Twelfth Intern. Conf. on Machine Learning, pp. 252–260 (2014)
11. Doshi, P., Goodwin, R., Akkiraju, R., Verma, K.: Dynamic workflow composition using markov decision processes. International Journal of Web Services Research **2**, 1–17 (2005)
12. Dutreilh, X., Kirgizov, S., Melekhova, O., Malenfant, J., Rivierre, N., Truck, I.: Using reinforcement learning for autonomic resource allocation in clouds: towards a fully automated workflow. In: The Seventh International Conference on Autonomic and Autonomous Systems, ICAS 2011, pp. 67–74 (2011)
13. Fadell, A., Rogers, M., Satterthwaite, E., Smith, I., Warren, D., Palmer, J., Honjo, S., Erickson, G., Dutra, J., Fiennes, H.: User-friendly, network connected learning thermostat and related systems and methods, July 4, 2013. https://www.google.fr/patents/US20130173064. uS Patent App. 13/656,189
14. Grzes, M., Kudenko, D.: Learning shaping rewards in model-based reinforcement learning. In: Proc. AAMAS 2009 Workshop on Adaptive Learning Agents, vol. 115 (2009)
15. Karray, F.O., De Silva, C.W.: Soft computing and intelligent systems design: theory, tools, and applications. Pearson Education (2004)
16. Nau, D., Ghallab, M., Traverso, P.: Automated Planning: Theory & Practice. Morgan Kaufmann Publishers Inc., San Francisco (2004)
17. Russell, S., Norvig, P., Canny, J., Malik, J., Edwards, D.: Artificial intelligence: a modern approach, vol. 2. Prentice hall Englewood Cliffs, NJ (1995)
18. Scott, J., Bernheim Brush, A., Krumm, J., Meyers, B., Hazas, M., Hodges, S., Villar, N.: Preheat: controlling home heating using occupancy prediction. In: Proceedings of the 13th International Conference on Ubiquitous Computing, pp. 281–290. ACM (2011)
19. Spiegelhalter, D.J., Dawid, A.P., Lauritzen, S.L., Cowell, R.G.: Bayesian analysis in expert systems. Statistical science, 219–247 (1993)
20. Strens, M.: A bayesian framework for reinforcement learning, pp. 943–950 (2000)
21. Tesauro, G.: Temporal difference learning and td-gammon. Communications of the ACM **38**(3), 58–68 (1995)
22. Tesauro, G., Kephart, J.O.: Pricing in agent economies using multi-agent q-learning. Autonomous Agents and Multi-Agent Systems **5**(3), 289–304 (2002)
23. Watkins, C.: Learning from Delayed Rewards. Ph.D. thesis, University of Cambridge, England (1989)
24. Wiering, M.: Multi-agent reinforcement learning for traffic light control. In: ICML, pp. 1151–1158 (2000)