Data-Driven User-Aware HVAC Scheduling

Daniel Petrov, Rakan Alseghayer, Daniel Mossé, Panos K. Chrysanthis
Department of Computer Science
University of Pittsburgh
{dpetrov, ralseghayer, mosse, panos}@cs.pitt.edu

***Abstract*—HVAC (Heat, Ventilation, Air Conditioning) systems account for significant amount of energy spent in residential and commercial buildings. Improved wall and window insulation, energy efficient bulbs as well as building design that facilitates a more optimal usage of the thermally conditioned air within a building, are amongst some of the measures taken to address the high usage of energy for space conditioning. In this paper we address a main issue that affects the energy consumption for heating and cooling of buildings, namely the duty cycle of the furnaces / air-conditioners. We propose D-DUAL, a 3-fold scheduling mechanism that builds on multiple variable linear regression model. Our scheduler minimizes the duty cycle and does not impact users' comfort. Our experimental evaluation shows that our proposed approach saves up to 49% energy, compared to commodity HVAC systems.**

***Index Terms*—IoT; HVAC; scheduling; smart home; energy savings; Internet of Things;**

I. Introduction

Motivation The energy consumption of private houses, commercial and public buildings has been increasing. The biggest amount of energy consumed in the US is for space heating and cooling of residential buildings–47.7% in 2009 [1]. This induces higher costs and calls for enabling new power plants that harm the environment [2]. To mitigate the negative impact on the environment it is imperative to optimize the energy consumption in residential buildings.

Many buildings in the US are equipped with thermostats control the temperature and sensors to detect the presence/absence of humans. Such "smart" buildings have systems in place to control the lighting as well. Recently, buildings are built with sensors and actuators that allow fine-grain control of the temperature at the room level and the duty cycles of the lighting. Moreover, the emergence of Internet of Things (IoT) enabled new technologies that facilitate increased autonomy in space conditioning and lightning–smartphone-based geofencing, as well as connected thermostats, power plugs, and light bulbs aim to improve quality of life [3]–[6].

Given a building that is enabled for fine-grain control of the temperature, the system can open or close the vents whenever necessary whenever the temperature in a given room diverges from the user requested one. Further, the (smart) thermostat or controller will command the HVAC unit to turn on at the appropriate time to regulate the temperature in the room. To the best of our knowledge, all proposed space conditioning solutions fall short to address a serious concern: *the amount of energy lost in turning the HVAC system on and off to regulate temperatures on a per-room basis.*

TABLE I
Energy Saving Approaches Using HVAC Scheduling

	Non-HVAC Scheduling	HVAC Scheduling
Non-ML Solution	[7], [8]	Naïve
ML Solution	[9]–[17]	D-DUAL

Approach In this paper, we propose an IoT solution that schedules the duty cycles of HVAC systems intelligently for energy reduction while meeting users' comfort requirements for target temperature, on a per-room basis, in residential buildings. Our solution, called D-DUAL, takes the desired temperature along with the maximum time the user expects for the temperature to be regulated (which we call *deadline*). Its innovation is that D-DUAL combines scheduling and regression techniques. The former optimizes HVAC duty cycles and the latter predicts the time needed to reach the desired temperature for each request. The predictive model is based on multiple linear regression (MLR). All requests and data sensor readings from all rooms are delivered to a "smart" gateway which runs the predictive model on the sensor readings. The gateway takes the deadlines for all requests from different rooms, as well as the respective predicted time values and prepares a schedule that controls the duty cycle of the HVAC system to minimize energy while maintaining users' comfort.

D-DUAL is a lightweight computational solution that can run the "smart" gateway on a real-life IoT "hub"–which is an integral part of many IoT deployments to this day. It is our goal to keep the computations locally into the hub and avoid exposure of users' data to additional privacy and security concerns when data is transferred to cloud. Furthermore, our solution is indifferent to the underlying IoT infrastructure. The information it operates on is fed into the gateway from the "hub" that receives sensor readings and user requests. The "hub" may be connected to wired, wireless or ad-hoc networks in order to receive data from sensors and users. Often these hubs are deployed on Raspberry Pi computers. In matter of fact we tested D-DUAL on Raspberry Pi.

Even though ML has been used in schemes to save energy in HVAC [9]–[11], none of these existing solutions aimed in optimizing the HVAC scheduling (for examples see Table I). These solutions used various ML algorithms for prediction as appropriate. In our case, as discussed below, we selected MLR for its relative simplicity of implementation, highly accurate results, and modest computational and memory footprint requirements.

Contributions

- We propose D-DUAL that combined three scheduling techniques, namely shortest job first, elevator and latest deadline, with a regression prediction model to reduce the loss of energy due to the duty cycle of the HVAC system.
- We develop a thermal energy exchange function that drives the MLR based prediction of time needed to reach a specified temperature.
- We show that the scheduling of space conditioning of rooms in residential buildings can be modeled as one of six *canonical cases*.
- We conduct an experimental evaluation on Raspberry Pi that shows that our proposed approach reduces the energy consumption for space conditioning by up to 49%.
- We also demonstrate via our evaluation the applicability of our solution for IoT systems as it can be deployed on low cost hardware, such as Raspberry Pi Zero, and produce results in real time - within 1 second.

Outline In Section II we set up the stage for our solution by providing the necessary definitions of the system model and the multiple linear regression. The solution and its two integral parts, namely, the scheduler and the *thermal energy exchange function* and *sliding window* that are used into MLR, are discussed in Section III. Experimental evaluation on a Raspberry Pi implementation is provided in Section IV. Related work is discussed in Section V and conclusions are provided in Section VI.

II. System Model and Preliminaries

In this section, we discuss our system model of a residential building, the optimization objective of our work and review MLR that we use with our thermal energy exchange function.

A room has direct space conditioning capabilities if there is a vent installed in the room, and that vent is connected to the controller that is in charge of space conditioning (of part of) the building. Each such room is equipped with a self contained sensing unit whose measurements are denoted $\{x_{ij}\}$, whereby i denotes the sample number and j denotes the sensor that generated the measurement.

Definition 1: (*Window of measurements*) A window w is a vector of n consecutive measurements of the sensors, ordered in time. The oldest one is at time $t-w$ and the most recent one is at time t. Each measurement contains the values from all available sensors that play role in the thermal energy exchange function used in the multiple linear regression.

Definition 2: (*Request*) A tuple

$$u_i(i, ts_{curr}, temp_{target}, ts_{target}) \tag{1}$$

is called a *request*, whereby ts_{curr} is the timestamp that marks when the request was generated in room i, $temp_{target}$ is the desired temperature to be achieved for this room, ts_{target} is a moment in the future by which the desired temperatures should be reached (the *"deadline"*).

Definition 3: (*Objective criterion*) Given the set Q_{curr} of current sensor readings for all rooms i, $1 \leq i \leq m$, the previous window w's sets of sensor readings $Q_{curr-1}, Q_{curre-2}, ..., Q_{curr-w}$, and the set R of requests u_i, generate a schedule S that minimizes the duty cycle of the furnace/AC and achieves the requested target temperatures $temp_{i,target}$ by the users' deadlines $ts_{i,target}$ for each room i in a house.

The schedule should be produced in near real time, whereby *n*ear real time stands for an amount of time that is short enough to provide sufficient time for the nearest deadline to be met. Given the objective criterion, our solution is indifferent to the underlying infrastructure that feed data into it. It is equally suitable for IoT deployments that have wired, Wi-Fi or ad-hoc communication networks. Moreover, ideally the solution will carry out the necessary computations locally to the IoT deployment and thus will not expose users' data to additional privacy and security concerns by pushing data onto clouds.

A. Multiple Linear Regression

Multiple linear regression (MLR) models the relationship between two or more explanatory variables x_i, δ_t and a response variable g by fitting a linear equation to observed data. Every value of the independent variable x_i is associated with a value of the dependent variable g. The population regression line for p explanatory variables $x_1, x_2, ..., x_p$ is defined as

$$\mu_g = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + ... + \beta_p x_p \tag{2}$$

This line describes how the mean response μ_g changes with the explanatory variables. The observed values for g vary about their means μ_g and are assumed to have the same standard deviation σ. The fitted values $b_0, b_1, ..., b_p$ estimate the parameters $\beta_0, \beta_1, ..., \beta_p$ of the population regression line.

Since the observed values for g vary about their means μ_g, the multiple regression model includes a term for this variation. Mathematically, the model is expressed as

$$\mathit{DATA} = \mathit{FIT} + \mathit{RESIDUAL} \tag{3}$$

where "FIT" represents the expression for g from Eq. 2.

$$\beta_0 + \beta_1 x_1 + \beta_2 x_2 + ... + \beta_p x_p \tag{4}$$

The "RESIDUAL" term represents the deviations of the observed values g from their means μ_g, which are normally distributed with mean 0 and variance σ. The notation for the MLR model deviations is ε.

Definition 4: The model for multiple linear regression, given n observations, is

$$g_i = \beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + ... + \beta_p x_{ip} + \varepsilon_i \tag{5}$$

for $i = 1, 2, ..., n$. In the least-squares model, the best-fitting line for the observed data is calculated by minimizing the sum of the squares of the deviations from each data point to the line (if a point lies on the fitted line exactly, then its deviation is 0). Because the deviations are first squared, then summed, there are no cancellations between positive and negative values.

The values fit by the expression $b_0+b_1x_{i1}+b_2x_{i2}+...+b_px_{ip}$ are denoted $\hat{g}_i$ and the residuals e_i are equal to $g_i - \hat{g}_i$, the difference between the observed and fitted values. The sum of the residuals is equal to zero.

The variance σ^2 may be estimated by

$$\sigma^2 = \frac{\Sigma e_i^2}{n-p-1} \tag{6}$$

also known as the mean-squared error (or MSE) [12]–[15].

MLR is used in our solution in a *sliding window* model, whereby we take a number of consecutive sensor readings for each room $(x_{11}, x_{12}, ..., x_{ij}, ..., x_{np})$ of the n rooms in the building and run regression on them to derive the coefficients β_i of $g(x_1, .., x_p, \delta_t)$. When new sensor readings are available, we drop the oldest readings and add the newest, ordered by time. The function g for each room is a building block for our scheduler, as discussed in the next section.

III. D-DUAL Solution

Our D-DUAL (Data-Driven User-Aware scheduLing) solution is depicted in Figure 1 and it consists of two integral elements: a prediction mechanism (regression) and scheduling mechanism–thus the name DUAL, i.e., "consisting of two".

A regression technique and a 3-fold scheduling mechanism, i.e., based on three fundamental scheduling mechanisms, are interwoven to generate optimal furnace/AC scheduling that minimizes the energy consumption and keeps the temperature in all rooms within the comfort range, as defined by the user requests for each room.

The intuition is to predict what amount of thermally conditioned air is needed for a room to reach the temperature desired by its users. Once this information is available, it is translated into a period of time in which the vent in the room should be open and air should be blown through it. This prediction for each room is used to make a schedule that minimizes the duty cycle of the HVAC system and reaches the target temperatures in all rooms. We have considered a number of ML techniques for our solution before adopting MLR. Specifically, we assume that historical data is available, which ruled out *unsupervised* and *reinforcement learning* techniques. Furthermore, in the domain of *supervised learning* techniques, we found most of *classification techniques* such as *logistic regression*, *linear discriminant analysis*, *classification and regression trees*, *naive Bayes*, *K-nearest neighbors*, *learning vector quantization and artificial neural networks*, as well as *support vector machines* unsuitable for our usecase. Some of them provide means for *binary classification only*, others have significant computational complexity and neither is compatible with our *thermal energy exchange function*. Moreover, we researched the usage of *autoregression* in our solution. *AUtoregression* is a single dimension time series predictive analysis technique. Therefore, we could either use it to predict the time needed to change the temperature by a unit of temperature or to predict the temperature change over a unit of time. The former limits our solution to using temperature sensor reading only and oversees other factors that affect temperature change. The latter requires additional computational model that calculates the prediction that is fed into the scheduler. We picked MLR because of its simplicity as it does not require tuning parameters such as in LASSO regression and Ridge regression to improve its accuracy in the case of small windows of readings as confirmed by our experimental evaluation (see Section II-A).

A. Prediction Model

We define a temperature change function g that is based upon our *thermal energy exchange* function f. The former is used in MLR-based prediction model (see Section II-A) to derive the amount of time needed to blow thermally conditioned air in a room to reach the desired temperature. Whenever a new sensor reading is received by the algorithm, the *recalculateCoefficients()* primitive is executed (see Lines 3-6, Algorithm 1). That slides the window for the room which the reading came from and MLR is run to derive the updated coefficients β_i. The regression coefficients are recalculated to maintain high accuracy for the predictions.

Definition 5: (*Thermal energy exchange*) Given p of the residential building's sensors installed in a room to measure the factors that affect the change of the temperature in that room, there is a linear function:

$$f(x_1, x_2, ..., x_p) = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + ... + \beta_p x_p \tag{7}$$

that measures the *thermal energy exchange* at a given unit of time t. The x_i are the measurements at time t, read from the sensors i, and β_i are the coefficients of the function, $1 \leq i \leq p$.

The intuition here for the thermal energy exchange is that the factors dictate if the temperature in the room will keep increasing or decreasing over a span of time, given the current values of sensors' readings.

Definition 6: (*Temperature change*) Given function $f(x_1, x_2, ..., x_p)$, there is a function

$$g(x_1, x_2, ..., x_p, \delta_t) = f(x_1, x_2, ..., x_p) \times \delta_t \tag{8}$$

that calculates the temperature change (in degrees) in the room, if the sensor readings $x_1, x_2, ..., x_p$ do not change for a period of time δ_t. Here, again x_i and δ_t are variables and β_i are the coefficients of the function, $1 \leq i \leq p$.

We use $g(x_1, x_2, ..., x_p, \delta_t)$ in our solution to calculate the amount of time for which terminally conditioned air should be blown into a room, so that room reaches a desired temperature. We call the variables x_i and δ_t *explanatory variables* and g *response variable* in the MLR.

B. D-DUAL Scheduler

The D-DUAL scheduling algorithm (see Lines 7-11, Algorithm 1) takes two types of input, namely new sensor readings x_{ij} and user requests, as defined in Eq. (8).

When a new request is received, it is parsed in the *parseRequest()* primitive. The *useCoefficientsToDeriveTime()* primitive is executed to derive the expected amount of time needed to reach the temperature for each request, given the last known sensor measurements for that room. When all

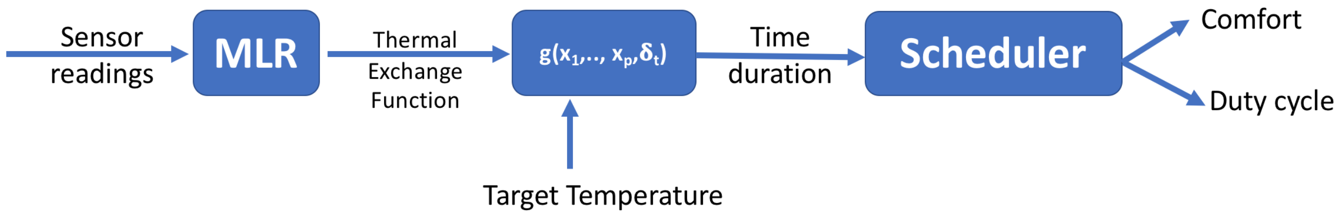

Fig. 1. D-DUAL Solution

Algorithm 1 D-DUAL

```
1:  procedure D-DUALSCHEDULER( )
2:      while 1 do
3:          if $newData\ x_{ij}\ is\ available$ then
4:              $slideWindow()$;
5:              $recalculateRegressionCoefficients(x_{ij})$;
6:          end if
7:          if $newRequests\ are\ available$ then
8:              $parseRequest()$;
9:              $useCoefficientsToDeriveTime()$;
10:             $recalculateSchedule()$;
11:         end if
12:     end while
13: end procedure
```

predictions are in place, a schedule is generated, adhering to the objective of the solution, as defined in Section 3.

The *recalculateSchedule()* primitive is based upon *Shortest Job First*, *Elevator Algorithm* and *Latest Deadline* scheduling principles. In case that all requests are for changing the temperature in the same direction (i.e., all rooms should be warmed up by the furnace; the case for cooling by AC is symmetric), the scheduling is trivial. The Elevator algorithm principle is employed in the sense that if the furnace is running already, it keeps running until all requests are fulfilled. If it is not running, the Latest Deadline approach is taken to batch requests and allow for new requests to be accommodated. The time windows needed to reach the desired temperature in each room are aligned in a way that minimizes the furnace duty cycle. Moreover, the furnace works for the amount of time equal to the longest of all predicted time frames. For the case when some rooms need heating and others need cooling, the requests are split into 2 subsets–those that require heating and those that require cooling. The aforementioned steps for each subset are executed independently of the other subset. The Shortest Job First principle is used as a tie breaker: the shorter of the two time frames is executed first.

C. Canonical Scheduling Cases

In this subsection we show that there is a finite number of scheduling cases that the scheduler should handle when running our *D-DUAL Solution* that aims to minimize the energy consumption for space conditioning of residential buildings. We use mathematical induction to show that all possible scheduling combinations fall into one of the cases that are discussed in this subsection. We run the induction for the number of rooms in the building.

Base Case: The basic case is the building with two room, $n = 2$. All possible combinations for 2 rooms are depicted in Figure 2. The case when the temperature does not have to be changed implies that the target temperature is achieved already and such rooms can be ignored. There are 12 cases on the diagram, labeled a to l. Each case has 2 lines, one for each room. The cases that require cooling are depicted in blue (dotted line) and the cases that require heating are depicted in red (solid line). A longer line depicts longer amount of time for which thermally conditioned air should be blown into the room. Often the rooms require different amounts of conditioned air to be provided in order to reach their desired temperatures. We depicted the case when the request that has arrived earlier also requires more air (and thus time) to reach the goal. The opposite case when the shorter request has an earlier arrival time, is symmetric. We did not depict it due to space limitations.

Given the intervals for two rooms there are three cases to be considered: (i) the intervals for the two rooms no overlap, (ii) the intervals overlap complete (i.e., one contains the other one) and (iii) partial overlap, depicted in Figures 2(a), 2(b) and 2(c), respectively. In these three cases the temperature in both rooms should be decreased. The three cases when both rooms should be heated follows on Figures 2(d), 2(e) and 2(f). The mixture of heating and cooling is depicted in Figures 2(g) to 2(l). The cases when the cooling predeceases the heating arrival is depicted in Figures 2(g), 2(h) and 2(i), and the opposite case in Figures 2(j), 2(k) and 2(l). Moreover, the temperatures in all rooms may need to be adjusted in the same direction (i.e., all need to be cooled down or all need to be warmed up). If the temperature in all rooms is expected to be adjusted in the same direction, there is no difference from scheduling point of view if all rooms require heating or cooling. This effectively implies that Case d. is identical to Case a, Case e to Case b and Case f to Case c. The other possibility is to have a mixture of heating and cooling. Similarly, the order of arrivals for cooling and heating when one room requires heating and the other cooling, do not make a different from scheduling point of view. It is to be noted that Cases j, k and l are identical to Cases g, h and i, respectively. We derive the conclusion that there are six base cases for scheduling and we

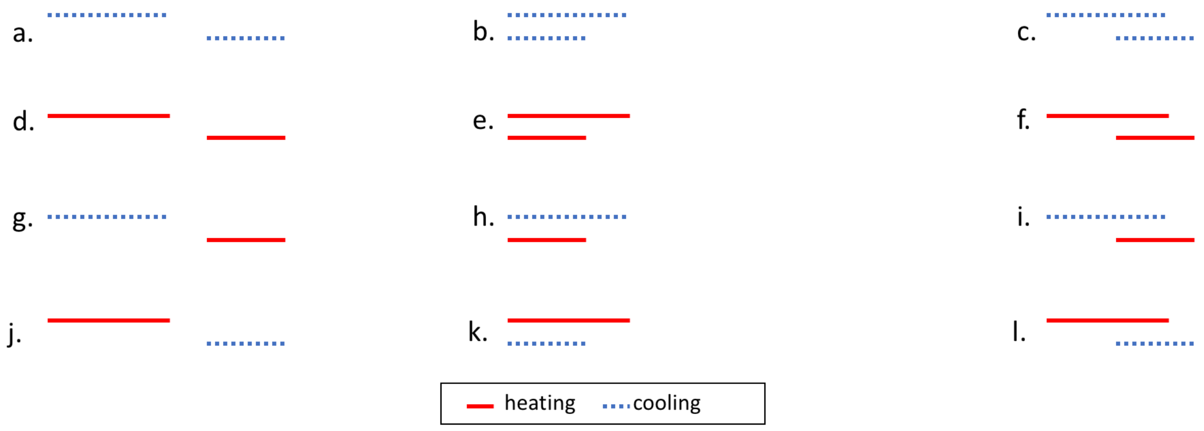

Fig. 2. Canonical cases for scheduling

refer to them as *canonical cases*–they are depicted in Figures 2(a), 2(b), 2(c), 2(g), 2(h) and 2(i).

Induction hypothesis: Assume that for some $n = k$ rooms, whereby k is a positive integer and $k > 2$ the time frames for space conditioning of the rooms in given house can be split into two groups. These groups cover one of the *Canonical cases*, presented in the *base case*. Further, each group can be split into two subgroups. These two subgroups cover one of the six canonical cases. This operation can be repeated recursively until each subgroup contains no more than two rooms.

Induction step: We will now show that for $n = k + 1$ the time frames for conditioning all rooms can be split in two groups that are in one of the *Canonical Cases*. If we arbitrarily take 1 room out of $k + 1$ rooms, we will have a set of k rooms, that we know that can be split into two groups in a way that builds a *canonical case*. If we consider one of these two groups arbitrarily and the room that has been taken out of the set earlier, as a group itself, they can be thought of as two groups. The time frame of the room that has no other room in its group, may not overlap with any of the time frames in the other group. This is a canonical case and it does not matter if the temperature in the single-grouped room should go in the same direction as all of the rooms from the other group, because our *canonical cases* cover both uni- and mix-direction cases. If the time frame for the single-grouped room overlaps completely with the time frames of the rooms from the other group, it can be added to their group and this will be other *canonical case*. Similarly, if he overlap of the time frame of that room with the other rooms is partial, this is another *canonical case*. This exhausts the possibilities for the relative position of the single-grouped time frame with respect to the other time frames. Hence by mathematical induction, the scheduling of n rooms, whereby $n \geq 2$, falls into one of the six *canonical cases*. $\square$

IV. Experiments and Analysis

In this section, we present the experimental evaluation results of our solution.

A. Experimental Framework

Testbed We implemented our solution and its algorithm in Java 1.8. We ran the experiments on a Raspberry Pi Zero W with 1 ARM CPU, running at 1GHz, 512MB of RAM memory, and 64GB MicroSDEX micro SD card. The operating system used was Raspbian Stretch Lite, based on Debian 9.

Metrics We evaluated the performance of the algorithm in terms of *prediction accuracy*, *wall-clock time* and *energy decrease*.

Prediction Accuracy: We measured how accurate our regression model (MLR) is in predicting the time needed to reach provided target temperature. We measured, the average of the differences between the predicted values and the actual amount of time needed to reach target temperatures in the rooms in scope. The actual amount of time and the temperature to be reached are extracted from the real dataset. We also use prediction accuracy to assess the suitability of MLR.

Wall-Clock Time: We measure the scalability of our solution, when deployed on low cost hardware. It is measured as the amount of time it takes for the computer to run our algorithm with the number of sensor readings we experimented with.

Energy Decrease: This is our optimization criterion. This metric reflects how capable the algorithm is in detecting rooms that can be conditioned without violating the comfort requirements of the users. It shows the amount of energy spent as a percentage of the amount of energy spent to achieve the same goal with commodity naive system. Assuming that other factors contributing to energy consumption in HVAC systems are constant, the length of the duty cycle of HVAC systems can be translated to the energy spent, that is, the longer the HVAC works, the higher the amount of energy spent on space conditioning.

Dataset *HiberSense Historical Data* [18]: The dataset we used in our experiments consists of thousand measurements from the HVAC related data in one family house for three days, collected between 2018-02-01 and 2018-02-03. The house has two rooms on the first floor and two rooms on the second floor. Each floor has its own thermostat. The data, available for each room, contains the measurements for motion, voltage of the sensors battery, two different temperature measurements, humidity level, air pressure and light level. Information about the state of the vent in each room is available as well. Vents can be either open or closed. The vents have the same sensors except that they do not contain measurements for motion. Moreover the dataset contains the following reading for each thermostat: fan state of the HVAC, state of the HVAC, temperature set on the thermostat, override state of the thermostat, hold state and the method the data was collected (push / pull). Outside temperature has been collected too, once per hour. User preference levels have been collected as well: the minimum and the maximum temperature the user tolerates as well as the safety boundary temperatures, beyond which the health of the user is jeopardized. All the data is timestamped with precision to a second. The senors in each room reported new measurements whenever there was a difference in the value of at least one reading, compared to the last values sent, or if a 15 minutes time span has passed. The thermostats reported every 3 seconds.

B. Experiments & Experimental Results

We ran four experiments to measure the metrics described above. We ran each experiment 5 times and reported the average and standard deviation of the metrics we collected during the experimental evaluation.

Experiment 1: Prediction Accuracy (Figure 3)–In our first experiment, we measured the time difference between our prediction, using our regression prediction model with MLR, and the actual amount of time, needed to reach the target temperature. We select sliding windows of different sizes of consecutive sensor measurements in a given room and run the regression prediction model to produce the predicted time. We use 5 different sensors to feed the regression model, specifically the average of the 2 temperature readings from the IoT unit that does not control the vent, the pressure and humidity readings of the same unit, the external temperature and the state of the HVAC system, i.e., heating, cooling or off. Further, we take the most recent record of sensor readings for that particular room (which includes the temperature reading), replace the temperature reading with the temperature we want to reach. The temperature we want to reach is nothing but fetched from the proceeding record, and we derive the predicted time it takes for the target temperature to be reached. We then compare the predicted duration against the actual duration observed in the dataset. We then slide the window by 1 record and repeat the steps until we exhaust the dataset.

We ran the experiment for seven different window lengths–specifically, 8, 16, 32, 64, 128, 256 and 512. Moreover, we ran it for each room independently. In Figure 3, we show the average of the five runs for all rooms and the standard deviation, whereby the standard deviation is calculated for the five runs for all rooms for a single window length. On the x-axis of the figure we have the different sliding

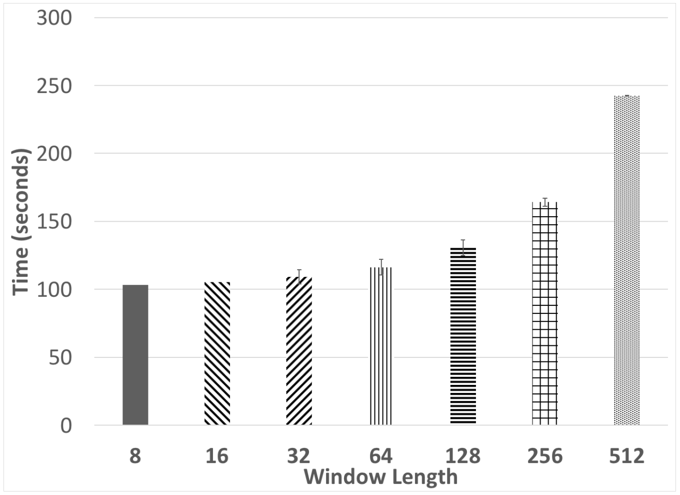

Fig. 3. Prediction accuracy with MLR for different window lengths for 4 rooms

window lengths. On the y-axis we show the average of the differences between the predicted values and the actual values. Our experiment showed that the accuracy of the regression library we used consistently decreased for windows of size 16 or larger. This is not surprising as the most recent values of sensor readings are the most relevant in affecting a current temperature. Furthermore, for our test dataset, window length of 8 readings predominantly provides the best results. We compared the predicted value for the amount of time needed to reach the target temperature to the ground truth and we observed that the predicted value consistently deviates by less than a minute for all rooms and for window length of 8. The experiment took about 67 seconds per window length when run on our testbed (Table II).

TABLE II
CLOCK TIME FOR EXPERIMENTS IN MILLISECONDS.

Experiment	Duration in msec
1st	945794 (15.7 min)
2nd	2000858 (33.34 min)
3rd	34684 (0.57 min)

Experiment 2: Regression Modules Comparison (Figure 4)–The goal of our second experiment was to quantitatively evaluate the suitability of MLR for our solution. In our second experiment, we measured the accuracy of the two most promising candidate regression modules, MLS and LASSO, to our problem. We ran the LASSO regression with multiple shrinkage parameter values (i.e., 0.1, 0.5, 1, 5, 10, 50, and 100). In Figure 4, our reported results are the average time difference between the predicted time to reach a certain temperature and the actual time taken to reach that temperature from the dataset. Those results are reported for the same 5 different window sizes (i.e., 8, 16, 32, 64, 128, 256 and 512). Clearly, for the data we have, MLR and LASSO are of the same accuracy when the window size is the smallest (size 8). However, when the window size is larger, the LASSO

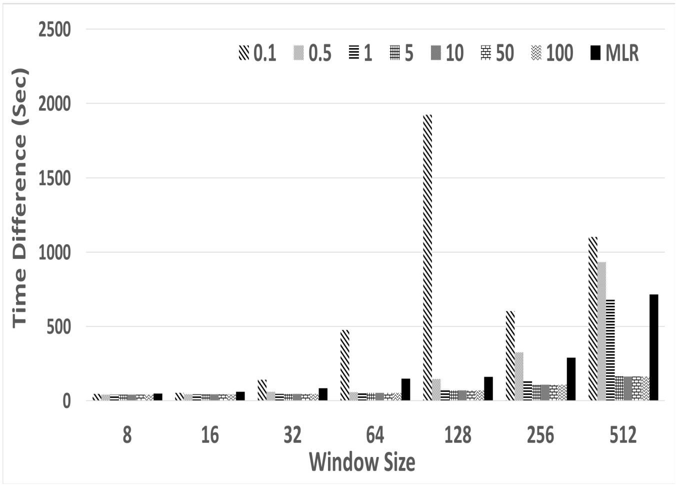

Fig. 4. Average time difference between predicted and actual time to reach a temperature for MLR and LASSO with different shrinkage parameter values

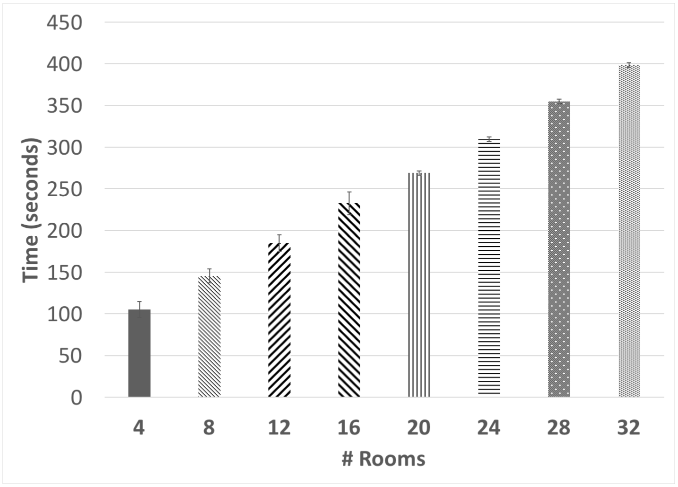

Fig. 5. Average time to run MLR for different number of rooms for sliding window length 8

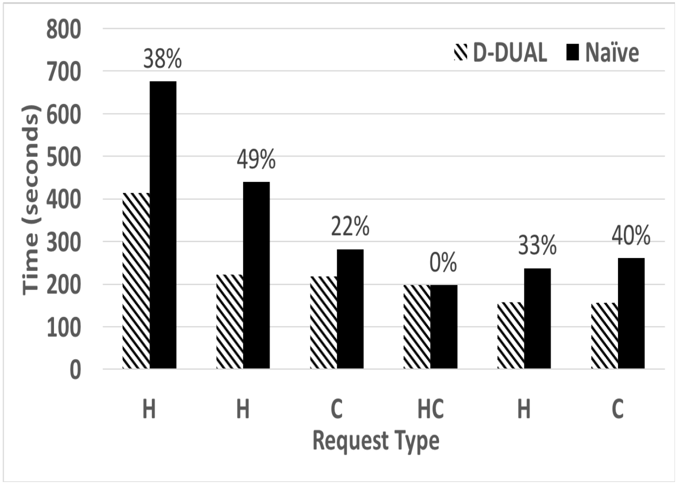

Fig. 6. Total durations of HVAC operation for *D-DUAL* and *Naive* HVAC for all canonical cases

produced more accurate results than MLR. However, the larger the window size, the lower the prediction accuracy is for both MLR and LASSO. This is consistent with the observation in Experiment 1, that small windows containing the most recent values of sensor readings are the most relevant in affecting a current temperature. By including more instances of sensor readings (i.e., larger window size) than what is recent, it would reduce the accuracy of both the modules prediction. Hence, using MLR with a small window size that yields the most accurate results and does not require any tuning parameters is a good first choice for D-DUAL.

Experiment 3: Scalability (Figure 5)–In our third experiment we study how our solution scales up with increase of the number of rooms, served by it. We measured the wall-clock time needed by our experimental testbed to run the regression from Experiment 1. We set a window of 8 consecutive measurements as input for the regression for each room. We set the number of rooms to 4, 8, 12, 16 20, 24, 28 and 32. Furthermore, we ran the experiment for all rooms and for all records, available for each room. The average amount of time to execute MLR is plotted in Figure 5. Our results show an increase of the time in relation to the increase of number of rooms. It takes less than 5 minutes for the solution to run regression for 32 rooms, 5000 times on average per room. It is to be noted, that regression for a sliding window of 8 can be calculated within 1 second or less (Table II) for up to 32 rooms simultaneously. This makes our solution, when deployed on Raspberry Pi Zero, suitable to manage space conditioning of single family houses and small office buildings.

Experiment 4: Energy Savings (Figure 6)–In our last experiment we measure the energy savings caused by our *D-DUAL* scheduling algorithm against a naive scheduler. The naive is a typical thermostat whose scheduler acts as soon as the request arrives. We pick two times a day from the three days data we have, and we derived scenarios from the dataset. Those resemble the cases discussed in Section III. This gives us a total of six scenarios that are diverse in the nature of their requests. In other words, each one of those scenarios consists of four requests (one for each room), that varies in the time of arrival and the cooling/heating request. We compare the performance of our *D-DUAL* scheduling against the naive scheduler by the predicted duration it takes the HVAC to fulfill the request. Figure 6 shows the total time needed to regulate (i.e., cool and/or heat) the temperature in the four rooms. The results show that our *D-DUAL* scheduling reduces the time to regulate the temperature in the four rooms by up to 49%. In only one case, *D-DUAL* performs the same as the naive scheduler.

V. Related Work

To the best of our knowledge, there is no prior work that combines ML and scheduling in an effort to optimize energy consumption in space conditioning of buildings only with local resources (i.e., not external network connectivity).

However, as summarized in Table I, ML techniques have been employed in HVAC optimization. One example of novel regression approach is presented in [11]. The paper discusses an incremental approach to calculate auto-regression, whereby the model is updated in $O(k)$ calculations at every step–a step is the next set of variables to be fed into the regression model. The auto-regression coefficients are calculated only once over a span of several steps. The computational cost to recompute the coefficients has complexity of $O(k^3)$. The alternative naive approach is to recalculate the coefficients at each step, having cost of $O(k^2)$. Moreover, the incremental update of the model and the sparse recalculation of coefficients raises a trade-off between computation cost and accuracy that is not thoroughly studied in the paper. Our approach recalculates the coefficients of the regression at each step. This ensures improved accuracy at computational cost with complexity of $O(k^2)$.

Room-level zoning of HVAC systems is tackled in [7], [8]. The three pillars of the work are HVAC dimensioning, occupancy prediction and thermal leakage. The first study, presented in the paper argues that the HVACs, installed into houses are under-dimensioned and thus their efficiency is decreased. Smaller HVAC installations and retrofitting of the buildings to room-level zoning is one of the directions to optimize energy consumption for space conditioning. Other dimension is improved insulation and the last one is "smart" resource allocation, whereby rooms that have no occupants are not conditioned. The occupancy detection is tackled further in [8]. Our work also tackles room-level zoning of HVAC systems and to certain extend we base our work on the studies in [7]. We assume that we can accurately detect room occupation. In contrast, our focus is on HVAC duty cycle scheduling that aims to decrease energy consumption without affecting the comfort of the users rather than sensing the presence of occupants in rooms and use that information to control vents.

The authors of [16], [17] used a model-predictive control mechanism that detects occupants in a room and uses that information to instruct the HVAC to compensate for the presence of people in the thermally conditioned room. The solution relies on semiparametric regression to calculate the temperature change in the room, given the presence of occupants. The energy gain they achieve is driven by the fact that the presence of people in a room increases the temperature in the room. Our work builds on top of that assumption and achieves energy saving from both an accurate *thermal energy exchange function* that can make use of occupants if such information is available, but also our novel 3-fold scheduling mechanism that optimizes the work cycles of the HVAC.

VI. Conclusions

In this paper, we presented our *D-DUAL* IoT solution that schedules the duty cycles of HVAC system intelligently to reduce energy consumption. The solution aims at minimizing the time of operation while meeting users' comfort requirements and specific requests for target temperature on a per-room basis in residential buildings.

Our solution combines scheduling and multiple linear regression techniques that take into consideration the temperature and the time window allowed to reach the desired temperature in each room, as well as sensor readings from the corresponding room. This information is delivered to a "smart" gateway, which prepares a schedule that controls the duty cycle of the HVAC system. We assumed that the users can tolerate the temperature changes in the rooms they submitted requests for a period of time until the deadlines that are part of their requests. Strictly, such changes do violate their comfort and hence in the future we plan to conduct a more thorough study on user comfort levels, specified explicitly by users.

Our experimental evaluation with real data showed that our approach achieves energy savings up to 49%, compared to the baseline commodity HVAC systems. Furthermore, we demonstrated that our computationally cheap solution can be deployed on low cost commodity hardware, such as Raspberry Pi, and it is capable of addressing the demands for HVAC control of real-world residential and commercial buildings.

References

[1] L. Doman, U.S. Energy Information Administration, "Use of Energy in the United States Explained, Energy Use in Commercial Buildings," www.eia.gov/energyexplained/index.cfm?page =us_energy_commercial, 2017, [Accessed 2018-02-19].

[2] ——, "EIA projects 48 increase in world energy consumption by 2040," www.eia.gov/todayinenergy/detail.php?id=26212, 2016, [2018-02-19].

[3] Nest Inc., "Nest Thermostats," nest.com/, 2016, [Accessed 2018-02-19].

[4] K. T. e. Nguyen, "Smartphone-based geofencing to ascertain hospitalizations," *Circulation: Cardiovascular Quality and Outcomes*, vol. 10, no. 3, 2017.

[5] Philips Inc., "Philips Hue Personal Wireless Lighting," www2.meethue.com/en-us/, 2018, [Accessed 2018-02-19].

[6] D-Link Inc., "D-Link Wi-Fi Smart Plug DSP-W215," us.dlink.com/products/connected-home/wi-fi-smart-plug/, 2018, [Accessed 2018-02-19].

[7] T. I. Sookoor, B. Holben, and K. Whitehouse, "Feasibility of retrofitting centralized HVAC systems for room-level zoning," in *IGSC*, 2012.

[8] E. Soltanaghaei and K. Whitehouse, "Practical occupancy detection for programmable and smart thermostats," *Applied Energy*, 2017.

[9] S. P. Kasiviswanathan, K. Nissim, and H. Jin, "Private incremental regression," in *ACM PODS*, 2017.

[10] P. Wang, R. Ge, X. Xiao, M. Zhou, and F. Zhou, "hmulab: A biomedical hybrid multi-label classifier based on multiple linear regression," *IEEE/ACM Trans. Comput. Biol. Bioinformatics*, vol. 14, no. 5, 2017.

[11] D. Mukherjee and S. Datta, "Incremental time series algorithms for IoT analytics: An example from autoregression," in *17th ICDCN*, 2016.

[12] S. Benmakrelouf, N. Mezghani, and N. Kara, "Towards the identification of players' profiles using game's data analysis based on regression model and clustering," in *ASONAM*, 2015.

[13] N. Jangid, S. Saha, A. Narasimhamurthy, and A. Mathur, "Computing the prestige of a journal: A revised multiple linear regression approach," in *WCI*, 2015.

[14] N. Graves and S. Newsam, "Visibility cameras: Where and how to look," in *MAED*, 2012.

[15] C. Mayfield, J. Neville, and S. Prabhakar, "Eracer: A database approach for statistical inference and data cleaning," in *SIGMOD*, 2010.

[16] A. Aswani, N. Master, J. Taneja, D. Culler, and C. Tomlin, "Reducing transient and steady state electricity consumption in HVAC using learning-based model-predictive control," *Proceedings of the IEEE*, vol. 100, 2012.

[17] A. Aswani, N. Master, J. Taneja, V. Smith, A. Krioukov, D. Culler, and C. Tomlin, "Identifying models of HVAC systems using semiparametric regression," in *ACC 2012*, 2012.

[18] HiberSense Inc., "Dataset," hibersense.com/, 2018, [Private communication; accessed 2018-02-19].