Scalable modeling of thermal dynamics in buildings using fuzzy rules for regression

Pablo Rodriguez-Mier*, Manuel Mucientes*, Alberto Bugarín*
*Centro Singular de Investigación en Tecnoloxías da Información (CiTIUS)
Universidade de Santiago de Compostela (USC), E-15782 Spain
{pablo.rodriguez.mier,manuel.mucientes,alberto.bugarin.diz}@usc.es

***Abstract*—The reduction of energy consumption in buildings is one of the goals to improve energy efficiency. One way to achieve energy savings in buildings is to develop intelligent control heating strategies that are able to reduce the power consumption by predicting the behavior of the thermal dynamics under different control schemes. One way to accomplish this is by means of learning fuzzy rules using the data collected from different sensors installed in buildings to generate regression models that are accurate and interpretable, so the generated models can be understood by the experts who approve the energy-saving schemes. However, one important issue is the generation of accurate knowledge bases of fuzzy rules for regression that can scale with the large amount of information generated by the many sensors installed in buildings, which will continue to grow in the coming years. For this purpose, in this paper we evaluate the scalability of two genetic fuzzy systems, FRULER and S-FRULER in the domain of thermal dynamics in buildings, using real data from a residential college at the USC.**

I. Introduction

Buildings account for 40% of the total energy consumption in the EU, according to European Directive 2010/31/EU on energy efficiency in buildings. Because of the expansion this sector is currently experiencing, a rise of that percentage will be inevitable. Therefore, it seems clear that the reduction of energy consumption and the use of energy from renewable sources in the building sector will play a key role in future measures to reduce emissions of greenhouse gases. One way to achieve energy savings in buildings is by reducing the total working hours of heating systems. However, a decrease in the total usage may lead to important decreases of indoor temperatures that can affect thermal comfort. In order to prevent this, automatic heating control systems must predict the future indoor temperature for a particular control policy in order to find the best strategy that minimizes power consumptions while keeping thermal comfort. Current methods for indoor temperature prediction [6] are mostly based on physical model simulations [19] and black-box machine learning methods [17], [9], [18], [20]. Physical models describe the building behavior by solving theoretical equations that describe to a certain precision the different dynamics and interactions between the variables. Although these methods are very powerful to simulate the different dynamics of a building, especially when there is no real data available, in general they are: 1) very time-consuming since they require many simulation hours, which prevents their application for predicting temperatures in small temporal windows; and 2) complex to formulate, since it is very difficult to produce a detailed model of a complex building, especially when there are many unknown factors that can affect the temperature dynamics. Machine learning models can overcome some of these limitations by learning the behavior from real data. However, current techniques, which are mostly black-box models based on neural networks, are hard to interpret and thus the interaction of the different variables of the building remains unknown. In this sense, the generation of accurate and interpretable models for thermal dynamics in smart buildings is fundamental 1) for modeling the thermal dynamics of the building to simulate the behaviour of the system to find better control strategies to reduce the energy consumption; and 2) to allow experts to interpret how the different variables of the system interact.

In this sense, fuzzy rule-based systems are well suited to this kind of applications thanks to their interpretability. However, there are some challenges associated with the automatic generation of rule-bases. Particularly, in Genetic Fuzzy Systems (GFS), the size of the problem has a huge influence in the performance of the algorithm [4], [7]. The rule bases learned suffer from exponential explosion as the number of variables increases and therefore the convergence time towards precise and simple models rises. Moreover, evolutionary algorithms are computationally expensive due to the large number of evaluations needed to reach convergence, and so the evaluation process to obtain the fitness may take a long time.

One way to cope with scalability issues from a Big Data perspective is to adopt the distributed computing paradigm for scaling GFS [11]. However, there is a lack of works –with only a few exceptions [5]– that use Big Data frameworks, such as Spark [22] or Hadoop [21] to deal with the scalability issues for regression problems. Concretely, the use of Spark is closely related to the success of Hadoop, which enables the processing of vast amounts of data in parallel on large clusters, usually implemented using the Hadoop Distributed File System. Spark adds to the Hadoop ecosystem the capability to use advanced data-flow computations with an improvement of in-memory computing and high-level functions that facilitate to build parallel applications.

It was not until recently that the use of GFS for solving large scale regression problems has started to attract attention in the field [2], [3], [13], [14]. However, the size of the training data used in these works is not large enough to be considered Big Data. Among the different approaches, FRULER [14] obtains Takagi-Sugeno-Kang 1-order (TSK-1) fuzzy rule bases with high accuracy and the lowest number of rules. Although the runtime of this approach is acceptable for the most simple datasets, it does not scale properly when solving large scale problems and may not converge to a good solution in reasonable time. These problems motivated the development S-FRULER, the distributed version of FRULER.

The main goal of our work is to address the problem of building accurate and interpretable models of thermal dynamics in buildings. One of the major issues is the large amount of data that will be generated in the following years, which will require the use of scalable learning algorithms – such as S-FRULER– to be able to improve the models over time with a reasonable computational effort. To do so, we use both FRULER and S-FRULER to learn accurate and simple TSK rules [15], and we compare them using the available current data on two different cases: 1) generation of indoor temperature models for different floors of the building and 2) generation of models for predicting the evolution of the temperature in a set of buffer tanks that are used store hot water. These models will be used later in the EU LIFE-OPERE project [1] to predict the behaviour of the building under different conditions in order to find new control strategies that lead to even further energy savings.

II. FRULER: Fuzzy RUle Learning through Evolution for Regression

FRULER (Fuzzy RUle Learning through Evolution for Regression) [15] is a novel GFS that obtains accurate and simple linguistic TSK-1 fuzzy rule base models for regression problems. FRULER (Fig. 1) is composed of a new instance selection method for regression, a novel multi-granularity fuzzy discretization of the input variables, and an evolutionary algorithm that uses a fast and scalable method with Elastic Net regularization to generate accurate and simple TSK-1 rules.

1) Instance selection.: The objective of the instance selection module is to reduce the variance of the models, focusing the generated rules on the representative examples. The instance selection method for regression is an improvement of the CCISR (Class Conditional Instance Selection for Regression) algorithm [12], which is an adaptation for regression of the instance selection method for classification CCIS (Class Conditional Instance Selection) [8].

2) Multi-granularity fuzzy discretization.: In a multi-granularity proposal, each granularity has a different fuzzy partition. The generation of the fuzzy linguistic labels can be divided into two stages. First, the variable must be discretized to obtain a set of split points C^g for each granularity g. Then, given the split points, the fuzzy labels can be defined for each granularity. In regression problems (TSK-1 in our case), the discretization process must search for the split point that minimizes the error when a linear model is applied to each of the resulting intervals.

3) Evolutionary algorithm.: The evolutionary algorithm learns a linguistic TSK model. The integration of the evolutionary algorithm with the preprocessing stage is as follows (Fig. 1):

- First, the instance selection process is executed over the training examples E_{tra} in order to obtain a subset of representative examples E_S.
- Then, the multi-granularity fuzzy discretization process obtains the fuzzy partitions for each input variable.
- Finally, the evolutionary algorithm searches for the best data base configuration using the obtained fuzzy partitions, generates the entire linguistic TSK rule base using E_S and evaluates the different rule bases using E_{tra}.

The chromosome is codified with a double coding scheme ($C = C_1 + C_2$). C_1 represents the granularity of each input variable. C_2 represents the lateral displacements of the split points of the input variables fuzzy partitions.

FRULER uses the Wang & Mendel algorithm to create the antecedent part of the rule base for each individual. The consequent part of the rules is learned using the Elastic Net method [23] in order to obtain the coefficients of the degree 1 polynomial for each rule. Elastic Net combines the ℓ_1 (Lasso regularization) and ℓ_2 (Ridge regularization) penalties of the Lasso and Ridge methods, minimizing the following equation:

$$\hat{\beta} = \arg\min_{\beta} ||Y - X \cdot \beta||^2 + \lambda \cdot \alpha \cdot \|\beta\|^2 + \lambda \cdot (1-\alpha) \cdot \|\beta\|_1 \quad (1)$$

where β is the coefficients vector, Y is the outputs vector, X is the inputs matrix, λ is the regularization parameter and α represents the trade-off between ℓ_1 and ℓ_2 penalization. In order to solve the minimization problem of Elastic Net (Eq. 1), we used Stochastic Gradient Descent (SGD).

The rule base is generated using only those examples in E_s. In this manner, those examples that are not representative are not taken into account, the method avoids the generation of too specific rules, and reduces the time needed to create the rule base. The fitness function is:

$$\mathit{fitness} = \mathit{MSE}(E_{tra}) = \frac{1}{2 \cdot |E|} \sum_{i=1}^{|E|} (F(x^i) - y^i)^2, \quad (2)$$

where E_{tra} is the full training dataset and $F(x^i)$ is the output obtained by the knowledge base for input x^i. Using all the examples for evaluation can be seen, in some way, as a validation process, as the rule base was constructed with a subset of them (E_S).

III. S-FRULER

S-FRULER [16] (Scalable Fuzzy Rule Learning through Evolution for Regression), is the distributed version of FRULER designed to improve the current scalability issues that hampers the use of FRULER with large-sized problems. To cope with these limitations, S-FRULER, instead of processing the entire dataset, divides the original problem into a set of smaller problems that are more tractable using a distributed approach (*Map phase*). Each of these divisions is then independently solved in the *Map phase* using the FRULER algorithm, as described in Sec. II. Finally, the solutions obtained in each

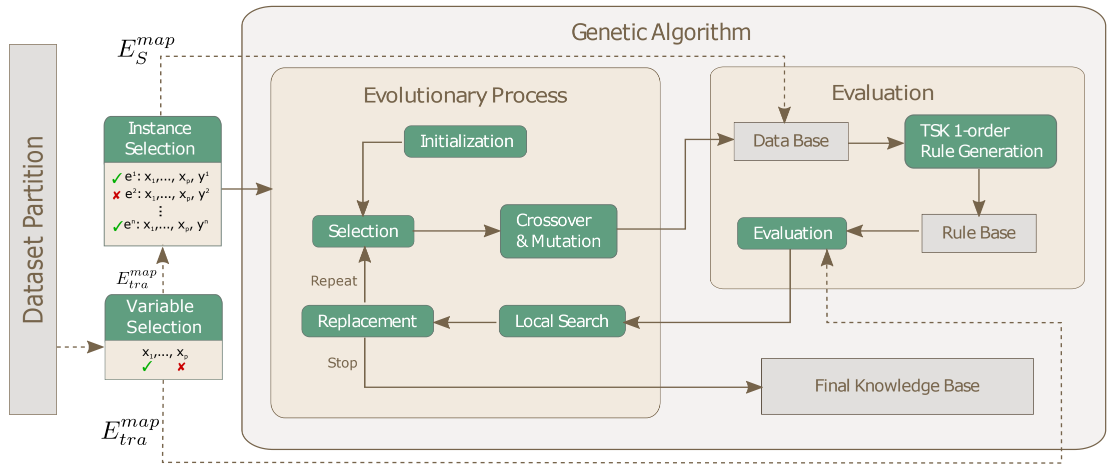

Fig. 1: FRULER architecture. Dashed lines indicate flow of datasets, dotted lines multigranularity information and solid lines represent process flow.

Map are combined in the *Aggregation phase* in order to obtain a final solution for the original problem.

The algorithm structure is shown in figure 2. The first step consists of a multi-granularity fuzzy discretization process that is performed using the whole training dataset. Then, the training dataset is splitted into n partitions during the *Map phase*. Those partitions generated during the *Map phase* corresponds with the tasks that are distributed as independent sets of processes to be processed in the worker nodes using *Apache Spark*. For each partition, only a subset of randomly selected variables is taken into account. Each partition is solved using FRULER, considering each partition as an independent problem, where only the instance selection and the genetic algorithm are executed. Finally, each independent solution for each sub-problem is combined in the *Aggregation phase*, where the missing variables that were not selected in some of the partitions are combined with the information of the other partitions to produce the final knowledge base.

IV. Modeling thermal dynamics in buildings

The main goal of the LIFE-OPERE European project [1] is to implement efficient management systems in both thermal and electrical energy grids in existing installations with large energy consumption. Particularly, in this project the study focuses on the residential college of Monte da Condesa, a building located at the University of Santiago de Compostela. Monte da Condesa comprises a set of centers that act as separate buildings, which nevertheless maintain thermal interaction through their conditioning circuits connected to a common cogeneration plant. The building is about 25,000 m^2 and reached in 2013 a total power consumption of 5,747 MWh. The set of all centers is supervised by a SCADA system that has 469 input and output variables that are associated with signals from the primary heating circuits and power consumption. Signals are collected in two different ways: synchronous (sync) and asynchronous (async). Synchronous signals are sequentially sampled at a fixed interval of 10 s, whereas asynchronous signals are registered by detecting a change of a value above a prefixed threshold. These signals include information about the indoor temperature of each floor, the outside temperature, the pumped water temperature of the heating systems, plus many other low level variables.

All these signals can be used not only to monitor and control the building but also to predict the behaviour of the system by observing its dynamics over time. Predicting the dynamics of the building is useful to perform a smart adjustment of the heating systems based on the predicted state of the building. In this context, three goals have been set out in this project: 1) prediction of indoor temperatures for each floor, taking into account future weather predictions to improve the current heating control scheme; and 2) modeling of the thermal dynamics in the buffer tanks, a set of five hot water storage systems for Domestic Hot Water (DHW) that are directly affected by the operation of an electric generation system.

The implementation of the system also requires the update of the models over time, using the new collected data. This implies that, every year, the training set increases by 8,760 hours of new data, which makes it necessary to use scalable machine learning techniques to prevent the whole system suffering from scalability issues over time as the training set grows.

To achieve these goals, in this work we propose a method that automatically learns scalable, accurate and interpretable non-linear models using S-FRULER.

A. Models for indoor temperature

In order to predict the indoor temperatures of each floor, we focus on the variables that may directly affect the temperature dynamics. These variables are represented in Fig. 3, which shows a high-level representation of the building. T_{in}^n, where $n = 0, \ldots, 5$, corresponds with the indoor temperature sensors of the building. Thus, there are 6 different sensors $(T_{in}^0, \ldots, T_{in}^5)$, one for each floor, which are the response

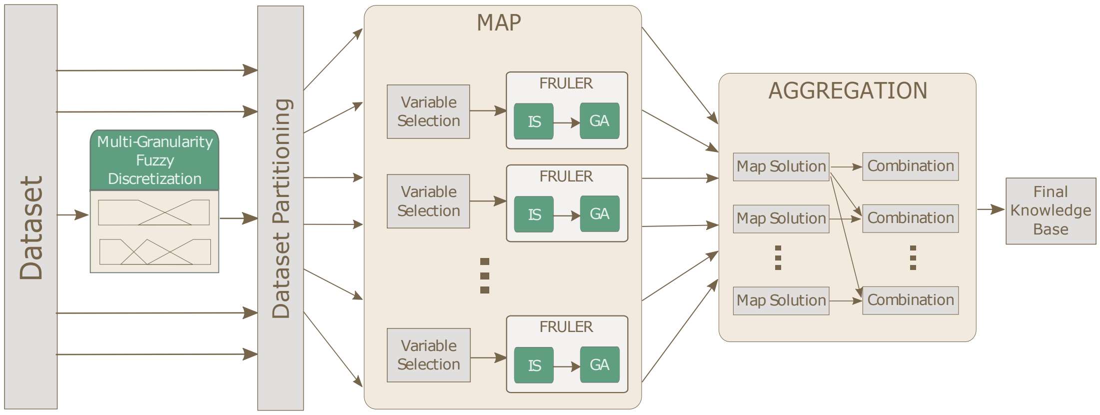

Fig. 2: S-FRULER architecture

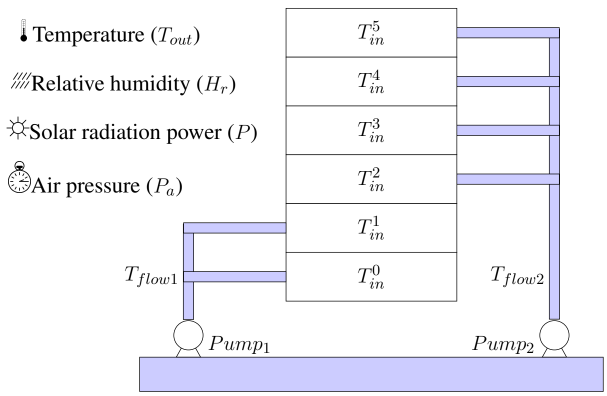

Fig. 3: Schema of the Monte da Condesa residential college with the related variables.

variables we aim to predict. Each floor is heated with hot water pumped from one of the two hot water pumps ($Pump_1$ and $Pump_2$). $Pump_1$ corresponds with the status of the pump of the heating system that feeds both floors 0 and 1, whereas $Pump_2$ corresponds with the status of the second pump that feeds the remaining floors. Note that, for the sake of clarity, in the following we will refer to $Pump$ instead of $Pump_1$ and $Pump_2$, where $Pump = Pump_1$ for floors 0 and 1 and $Pump = Pump_2$ for floors from 2 to 5.

In addition to these SCADA variables installed in the building, we also obtained the humidity (H_r), solar radiation power (P), and pressure (P_a) from *Santiago-EOAS*, a Meteogalicia [10] weather station situated approximately 100 meters from the building. These features give relevant information about weather conditions that may directly affect the indoor temperatures. Moreover, the temperature (T_{out}^{MS}), relative humidity (H_r^{MS}) and pressure (P_a^{MS}) predictions are obtained from *MeteoSIX*, a Galician numerical weather prediction service that provides hourly predictions from the current day to four days in ahead. MeteoSIX predictions provide information about future weather conditions at a given instant of time.

Synchronous measures were downsampled to 1 h bins and asynchronous measures were converted into time series by appling linear interpolation and 1 h resampling. To summarize, the selected signals, sampled at 1 h interval (t) are:

- $T_{in}^n(t)$: indoor temperature at t of floor n (°C, async).
- $T_{out}(t)$: outside temperature at t (°C, async).
- $Pump(t)$: binary status (1-on, 0-off) of the water heating pump at t (sync).
- $H_r(t)$: relative humidity (%, sync, Meteogalicia).
- $P(t)$: global solar radiation power (W/m^2, sync, Meteogalicia).
- $P_a(t)$: air pressure (hPa, sync, Meteogalicia).
- $T_{out}^{MS}(t)$: outdoor temperature prediction (°C, MeteoSIX).
- $H_r^{MS}(t)$: relative humidity prediction (%, MeteoSIX).
- $P_a^{MS}(t)$: air pressure prediction (hPa, MeteoSIX).

The variable $Pump$ is one of the most important features of the model, since pumping hot water to the building has a direct impact on both the indoor temperature and the energy consumption. Thus, this variable can be controlled to simulate different heating control schemes to maximize energy savings while keeping thermal comfort. For this purpose, we use the information of the previous 24 hours, but instead of using 24 binary features (1-Pump ON, 0-Pump OFF for each hour), we generate 4 features grouping the last 24 hours into 4 groups of 6 hours:

- $Pump^0(t)$: total ON hours from t to $t-6$.
- $Pump^1(t)$: total ON hours from $t-6$ to $t-12$
- $Pump^2(t)$: total ON hours from $t-12$ to $t-18$
- $Pump^3(t)$: total ON hours from $t-18$ to $t-24$.

We constructed a rule-based regression model F with S-FRULER to predict each variable response $\hat{T}_{in}^n(t,k), n \in [0,5]$ for different values of k (period lags), where $\hat{T}_{in}^n$ is the predicted indoor temperature on floor n at time instant $t+k$.

$$\begin{aligned}\hat{T}_{in}^n(t,k) = F[&T_{in}^n(t), T_{out}(t), H_r(t), P(t), P_a(t),\\ &T_{out}^{MS}(t+k), H_r^{MS}(t+k), P_a^{MS}(t+k),\\ &Pump^0(t+k), \ldots, Pump^3(t+k)]\end{aligned}$$

In order to train the models, several values of k could be set. In this case, $k = \{2, 4, 8, 16, 32, 64\}$h are proposed.

B. Buffer tanks temperature prediction

In addition to the gas-fired boilers installed in the building that are used for heating water, the heating system also has a cogeneration system that provides thermal energy to the buffer tanks and produces electricity. The residual heat produced by the engine is transferred to a set of five buffer tanks. These tanks are used to store the hot water not used during periods where the electrical engine is working, so that it can be used later during periods of high demand (basically, heating and DHW consumption). Predicting the thermal dynamics in the buffer tanks is important in order to draw a better schedule of the cogeneration operation.

Figure 4 shows the five buffer tanks. Each tank has three thermal sensors to monitor the water temperature at three different levels (lower, middle and upper positions). For this problem, only the upper and lower temperature is taken into account, since it is a reliable indicator of the available thermal energy in the tanks. Besides, the temperature in all tanks has a similar behaviour, so that we can consider T_{tank}^{up} as the average upper temperature and T_{tank}^{low} as the average lower temperature.

In addition to the weather features used for predicting the indoor temperature, we selected the following features to be included in the final model:

- $T_{tank}^{up|low}(t)$: average upper/lower temp. at t (°C, sync).
- $DHW(t)$: estimated 24 hours of domestic hot water consumption, using the values of the previous week as an estimation (m^3, async).
- $Cog(t)$: cogeneration system status (1-ON, 0-OFF), (binary, async).

$Cog(t)$, as in the case of the $Pump(t)$ variable used in the indoor temperature models, is the control variable that can be used to optimize the cogeneration scheme. $Cog(t)$ is also grouped into 4 groups of 6 hours ($Cog^0 \ldots Cog^3$). The predicted temperature in the tanks is modelled as:

$$\begin{aligned}\hat{T}_{tank}(t,k) = F[&T_{tank}(t), T_{out}(t), H_r(t), P(t), P_a(t),\\ &T_{out}^{MS}(t+k), H_r^{MS}(t+k), P_a^{MS}(t+k),\\ &DHW(t+k), Cog^0(t+k), \ldots, Cog^3(t+k)]\end{aligned}$$

V. Results

In this section we compare the performance of FRULER vs S-FRULER [16]. We collected a total of 8,760 hours of training data from 2016-02-01 to 2017-01-31 for the indoor temperature models, and a total of 6,161 hours from 2016-05-19 to 2017-01-31 for the buffer tank models. For each run, the dataset was randomly divided into training (80%) and test (20%). Both FRULER and S-FRULER have been used with the default settings. In the case of S-FRULER, models were generated using the multithread mode with 8 threads on an Intel® Core™ i7-3770 CPU at 3.40GHz.

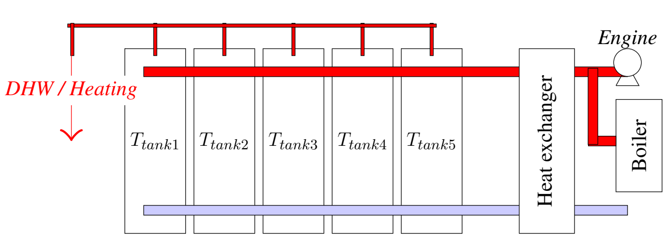

Fig. 4: Buffer tanks and cogeneration engine schema.

1) Indoor temperature models: We generated a total of 36 models, one for each $k \in \{2, 4, 8, 16, 32, 64\}h$ and for each floor of the building, with 12 input variables, using both FRULER and S-FRULER. Table I and II show the RMSE, number of rules, and total time for each approach for the models of the floor 0 and 1 respectively. We omitted the results of the remaining floors due to lack of space. We only include the RMSE of the test error since both training and test errors were very close and no overfitting problems have been identified.

TABLE I: Indoor temperature models for *floor 0*

	FRULER			S-FRULER		
k	RMSE	#Rules	Time (s)	RMSE	#Rules	Time (s)
2h	0.27	53	4781	0.32	6	687
4h	0.41	25	9074	0.45	11	708
8h	0.54	21	8247	0.69	5	581
16h	0.60	24	8681	0.78	4	699
32h	0.70	21	9408	0.85	2	698
64h	0.75	15	2159	0.96	6	724

TABLE II: Indoor temperature models for *floor 1*

	FRULER			S-FRULER		
k	RMSE	#Rules	Time (s)	RMSE	#Rules	Time (s)
2h	0.26	39	8432	0.33	6	913
4h	0.42	33	9122	0.49	9	810
8h	0.55	86	25212	0.72	6	786
16h	0.71	13	4552	0.78	4	826
32h	0.77	54	6234	0.87	6	800
64h	0.84	37	11967	1.01	16	839

As can be seen, FRULER always generates models with lower errors (≈20% lower RMSE) but at expenses of a higher computational cost and complexity. In the worst case, FRULER takes 7h to generate a model of the *floor 1* for $k = 8h$ (Table II), using a total of 86 rules, which clearly goes against the interpretability goal. Most part of this time is spent on the evaluation of the individuals by the EA (see [14] for more details). On contrast, S-FRULER is on average 12 times faster than FRULER using the same computer (all models were generated in less than 15 min) with rule bases that are 6 times smaller on average.

2) Buffer tank temperature models: In this case, we generated two types of models: one for the average temperature of the upper part of the tanks and the other for the lower part, using in total 13 input variables. Again, we observe a similar behavior as in the case of the indoor temperatures. Errors are again larger for the models generated with S-FRULER (≈

TABLE III: Lower tank temperature models

	FRULER			S-FRULER		
k	RMSE	#Rules	Time (s)	RMSE	#Rules	Time (s)
2h	2.36	3	1259	2.45	2	480
4h	2.72	6	2693	3.78	4	560
8h	3.69	16	3376	4.45	6	535
16h	3.89	16	2285	4.61	12	506
32h	4.72	23	2430	6.10	12	505
64h	4.96	48	3423	6.08	4	532

TABLE IV: Upper tank temperature models

	FRULER			S-FRULER		
k	RMSE	#Rules	Time (s)	RMSE	#Rules	Time (s)
2h	2.23	8	2333	2.27	6	448
4h	2.45	12	2450	3.16	6	464
8h	2.83	30	5704	4.31	4	456
16h	3.69	24	2721	4.57	6	440
32h	3.91	19	5974	5.32	3	430
64h	3.77	21	5174	4.98	6	485

25% larger), but using much smaller rule bases (4x smaller on average) and much faster (7x faster on average).

VI. Conclusion

In this paper we presented a novel approach for modeling the thermal dynamics of buildings using the information of different sensors to automatically generate a knowledge base of fuzzy rules for regression. To do so, we focus on Monte da Condesa, a building of the University of Santiago de Compostela with 469 sensors that provide information of the different parts of the system. One of the main issues is the generation of interpretable and accurate fuzzy models in reasonable time, given the large amount of data generated in the building, a problem that is going to grow year after year as more information is available. This requires the use of scalable techniques to be able to cope with the increase in complexity. For this purpose, we used S-FRULER, a distributed algorithm for learning fuzzy rules that can scale with the size of the problem, and we compare its performance against FRULER, the original non-distributed version of the algorithm.Results proved that S-FRULER clearly improves FRULER in terms of number of rules and runtime, obtaining rule bases 6 times smaller on average and an average speedup of 11.7, with only a 21% increase of the average RMSE.

Acknowledgment

This research was supported by the European Union LIFE programme (grant LIFE12 ENV/ES/001173). Also the following support is acknowledged: Spanish Ministry of Economy and Competitiveness (grants TIN2014-56633-C3-1-R and TIN2015-70308-REDT), Galician Ministry of Education, Culture and Universities (grants GRC2014/030 and accreditation 2016-2019, ED431G/08). These grants are co-funded by the European Regional Development Fund (ERDF/FEDER program).

References

[1] Life-OPERE, http://www.life-opere.org/, Last visited Jan 28th 2016.
[2] Alcalá, R., Alcalá-Fdez, J., Herrera, F., and Otero, J. Genetic learning of accurate and compact fuzzy rule based systems based on the 2-tuples linguistic representation. *International Journal of Approximate Reasoning 44*, 1 (2007), 45–64.
[3] Alcalá, R., Gacto, M. J., and Herrera, F. A fast and scalable multiobjective genetic fuzzy system for linguistic fuzzy modeling in high-dimensional regression problems. *IEEE Transactions on Fuzzy Systems 19*, 4 (2011), 666–681.
[4] Cordón, O., Herrera, F., Hoffmann, F., Magdalena, L., Cordon, O., Herrera, F., and Hoffmann, F. *Genetic fuzzy systems*. World Scientific Publishing Company Singapore, 2001.
[5] Fernández, A., López, V., del Jesus, M. J., and Herrera, F. Revisiting evolutionary fuzzy systems: Taxonomy, applications, new trends and challenges. *Knowledge-Based Systems* (2015).
[6] Foucquier, A., Robert, S., Suard, F., Stéphan, L., and Jay, A. State of the art in building modelling and energy performances prediction: A review. *Renewable and Sustainable Energy Reviews 23* (2013), 272–288.
[7] Herrera, F. Genetic fuzzy systems: taxonomy, current research trends and prospects. *Evolutionary Intelligence 1*, 1 (2008), 27–46.
[8] Marchiori, E. Class conditional nearest neighbor for large margin instance selection. *IEEE Transactions on Pattern Analysis and Machine Intelligence 32*, 2 (2010), 364–370.
[9] Mechaqrane, A., and Zouak, M. A comparison of linear and neural network ARX models applied to a prediction of the indoor temperature of a building. *Neural Computing and Applications 13*, 1 (2004), 32–37.
[10] Meteogalicia. Galician meteorological web page, http://www.meteogalicia.es/, Last visited Jan 28th 2016.
[11] Reyes-Ortiz, J. L., Oneto, L., and Anguita, D. Big data analytics in the cloud: Spark on Hadoop vs MPI/OpenMP on Beowulf. *Procedia Computer Science 53* (2015), 121–130.
[12] Rodríguez-Fdez, I., Mucientes, M., and Bugarín, A. An instance selection algorithm for regression and its application in variance reduction. In *Proceedings of the IEEE International Conference on Fuzzy Systems (FUZZ-IEEE)* (2013), pp. 1–8.
[13] Rodríguez-Fdez, I., Mucientes, M., and Bugarín, A. Reducing the complexity in genetic learning of accurate regression TSK rule-based systems. In *Proceedings of the 2015 IEEE International Conference on Fuzzy Systems (FUZZ-IEEE)* (2015), IEEE, pp. 1–8.
[14] Rodríguez-Fdez, I., Mucientes, M., and Bugarín, A. FRULER: Fuzzy rule learning through evolution for regression. *Information Sciences* (2016), doi:10.1016/j.ins.2016.03.012.
[15] Rodríguez-Fdez, I., Mucientes, M., and Bugarín, A. FRULER: Fuzzy rule learning through evolution for regression. *Information Sciences 354* (2016), 1–18.
[16] Rodríguez-Fdez, I., Mucientes, M., and Bugarín, A. S-FRULER: Scalable Fuzzy Rule Learning through Evolution for Regression. *Knowledge-Based Systems 110* (2016), 255 – 266.
[17] Ruano, A., Crispim, E., Conceicao, E., and Lúcio, M. M. Prediction of building's temperature using neural networks models. *Energy and Buildings 38*, 6 (2006), 682–694.
[18] Tao Lu, and Viljanen, M. Prediction of indoor temperature and relative humidity using neural network models: model comparison. *Neural Computing Applications 18*, 4 (2009), 345–57.
[19] Teodosiu, C., Hohota, R., Rusaouën, G., and Woloszyn, M. Numerical prediction of indoor air humidity and its effect on indoor environment. *Building and Environment 38*, 5 (2003), 655–664.
[20] Thomas, B., and Soleimani-Mohseni, M. Artificial neural network models for indoor temperature prediction: investigations in two buildings. *Neural Computing and Applications 16*, 1 (2006), 81–89.
[21] White, T. *Hadoop: The definitive guide*. O'Reilly Media, Inc., 2012.
[22] Zaharia, M., Chowdhury, M., Franklin, M. J., Shenker, S., and Stoica, I. Spark: cluster computing with working sets. In *Proceedings of the 2nd USENIX conference on Hot topics in cloud computing* (2010), vol. 10, p. 10.
[23] Zou, H., and Hastie, T. Regularization and variable selection via the elastic net. *Journal of the Royal Statistical Society 67*, 2 (2005), 301–320.