Applied Statistical Model Checking for a Sensor Behavior Analysis

Salim Chehida, Abdelhakim Baouya, Saddek Bensalem, Marius Bozga

Applied Statistical Model Checking for a Sensor Behavior Analysis

Salim Chehida[0000−0002−5070−2591], Abdelhakim Baouya[0000−0003−2182−7501], Saddek Bensalem[0000−0002−5753−2126], and Marius Bozga[0000−0003−4412−5684]

University Grenoble Alpes, CNRS, VERIMAG
F-38000,Grenoble, France
{salim.chehida, abdelhakim.baouya, saddek.bensalem,
marius.bozga}@univ-grenoble-alpes.fr

Abstract. The analysis of sensors' behavior becomes one of the essential challenges due to the growing use of these sensors for making a decision in IoT systems. The paper proposes an approach for a formal specification and analysis of such behavior starting from existing sensor traces. A model that embodies the sensor measurements over the time in the form of stochastic automata is built, then temporal properties are feed to Statistical Model Checker to simulate the learned model and to perform analysis. LTL properties are employed to predict sensors' readings in time and to check the conformity of sensed data with the sensor traces in order to detect any abnormal behavior.

Keywords: IoT · Sensor· Stochastic Automata · Statistical Model Checking · LTL · BIP

1 Introduction

Internet of Things (IoT) has become one of recent technology mostly used in various domains such as health and environmental monitoring [26], construction and energy management [22], smart vehicles [2], and buildings [7]. It consists of a collection of entities that interacts with users to fulfill a common goal. The sensor is a critical device in the IoT ecosystem that allows to measure the state information over time and monitor physical components. Data gathered from different sensors are used to make a decision and promote automation in IoT systems by providing efficient and intelligent services, whereas, corrupted data during transmission or malfunction of sensors, due to natural events or other causes can influence the correct operation of the entire system. Indeed, the massive increase of these issues with the growing number of deployed sensors push towards the sensors' behavior analysis by checking their sensed data.

The analysis of sensors' behavior and detecting the erroneous readings have attracted great attention. Many approaches have been proposed based on several methods such as statistical methods [30], probabilistic methods [28, 14], clustering-based methods [12] and prediction-based methods [25]. Governed by

the standard learning requirements, the approaches rely on the metadata and structure of the sensed data.

In this paper, we propose a model-based approach involving formal verification for sensor behavior analysis. Our approach aims to make the analysis process of sensed data rigorous, automatic, scalable, and meaningful. Figure 1 shows the steps of our approach. First, we start by collecting sensor traces and data preprocessing required to build an approximate model of the sensor behavior, then we apply formal verification techniques to analyze this model and then check if new measurements are compliant with the learned model.

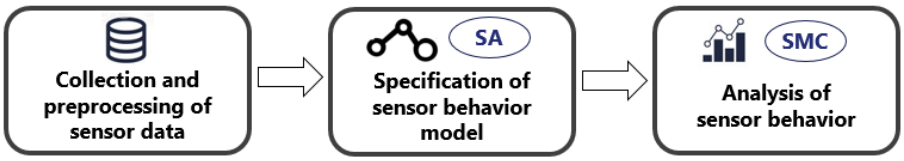

Fig. 1. Generic Approach for Sensor Behavior Analysis

Model checkers allow checking the conformity of a system model expressed in formal notation to a set of properties expressed in a logical language. In this study, we apply a type of model checkers called Statistical Model Checkers (SMC) to verify whether a sensor model expressed in Stochastic Automata (SA) satisfies a given logical property up to some probability, based on model simulations. We use quantitative properties expressed by Linear-time Temporal Logic (LTL) to predict the sensor readings in time and qualitative LTL properties to check the quality of sensed data and their compliance with the provided traces. Several SMC tools have been proposed such as PRISM-SMC [15], UPPAAL-SMC [8]. The BIP language [4] and *S*BIP [17] are used in this paper for behavior modeling and SMC analysis. We apply our approach to the industrial case study of the Cecebre dam in Spain, which is equipped with wireless sensors that measure the water contributions to the dam.

This paper is organized as follows: we build the sensor behavior model in Section 2. The analysis results of the sensor model will be presented in Section 3. Finally, we present related works in Section 4 and draw our conclusions in Section 5.

2 Sensor Behavior Model

In this work, we use BIP [1] (Behavior, Interaction, Priority), a component-based language for rigorous design of systems. In BIP, components are finite-state automata having transitions labeled with ports and states that denote control

[1] https://www-verimag.imag.fr/TOOLS/DCS/bip/doc/latest/html/index.html

locations (see Figure 4). We first start by data preprocessing and extraction of some statistical information needed to build the behavior models of sensors.

2.1 Data Preprocessing

In our case study, we consider three sensors deployed in the dam of *Cecebre* in the city of *la Coruna* in Spain. These sensors are used to measure the Water Height (WH), the Rain Precipitation (RP), and the Water output Flow (WF). As shown in Figure 2, the data collected from sensors are used to control the opening of the spillgate in order to ensure that the water does not reach a maximum level in the dam. The anomalous behavior of these sensors can influence the correct operation of the dam system. Our objective is to build formal models that specify the normal behavior of the sensors. These models will be used to control the sensors' readings and to detect any failure or anomaly.

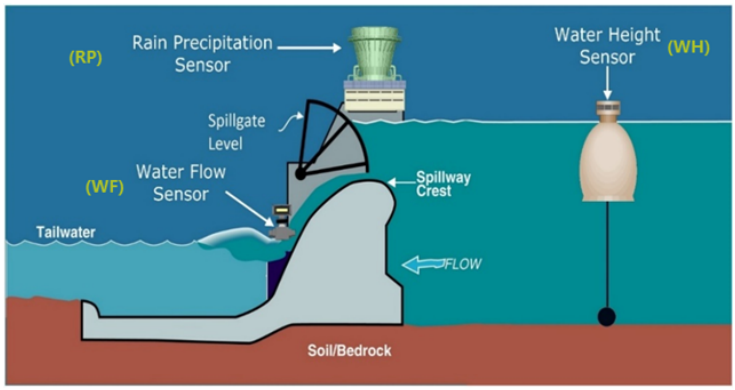

Fig. 2. Dam Infrastructure

A trace of time series data recorded by each sensor per day since 1989 to 2016 has been collected. We reorganized the original trace by creating a separate CSV file per sensor. The new file contains the sensor readings per day for 28 years. As shown in Figure 3, the data preprocessing is done in three steps:

1. Data cleaning: we use a filter to remove faulty sensors data. The filter deletes NaN data and data that not make sense, such as negative and inconsistent data.
2. Data discretization: we convert continuous (or quantitative) data into discrete (or qualitative) ones. The paper [27] presents the several methods proposed for time series data discretization. In this study, we use the EWD (Equal Width Discretization) method [9] because of its simplicity. It consists of mapping numerical values into predefined fixed intervals that have an equal-width. Each bin or level is associated with a distinct discrete value.

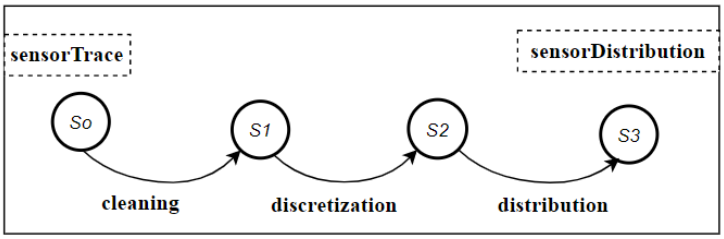

Fig. 3. Preprocessing of Sensors Data

In this work, we relied on data visualization using histograms to determine the number of levels. For the water height sensor, we use five levels for data discretization.

3. Generation of distribution: Once data was discretized, we extract some statistical information. We generate a sensor distribution file by counting the occurrence of each level of water height (WH_L) each day.

2.2 Specification of Sensor Model

Figure 4 presents a behavior model for the water height sensor expressed in the BIP language.

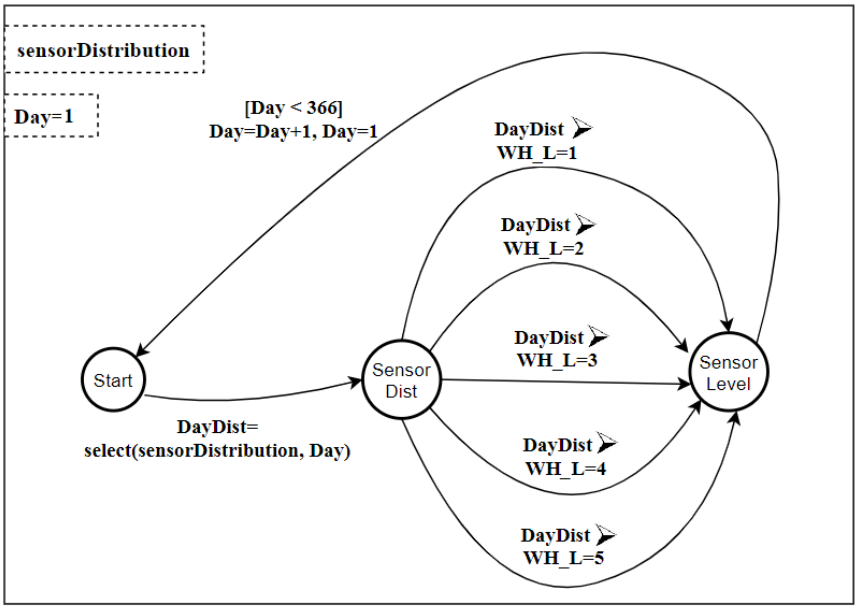

Fig. 4. Behavior Model of Water Height Sensor

BIP supports several formal modeling formalisms based on Discrete and Continuous Time Markov Chains (DTMC and CTMC) and Generalized Semi-Markov Process (GSMP). In this work, we use Stochastic Automata (SA) to express a behavior model of the sensors. The stochastic semantics is defined by variables based on the probability distributions. In the model of Figure 4, we select the day distribution based on `sensorDistribution` file generated in the previous section. According to this distribution, the water high level (1, 2, 3, 4 or 5) is defined.

The models that specify the behaviors of the other sensors (RP and WF) are defined using the same pattern as WH sensor model. Only the number of levels can change depending on the sensor data. Using these models, we can simulate and analyze the behavior of the different sensors for any period of the year.

3 Analysis of Sensor Behavior

*S*BIP framework[2] has a graphical user-interface permitting to edit, compile and simulate models, and automates the different statistical analysis. As shown in Figure 5, the input of the tool is a system model `S` expressed in BIP language like that of Figure 4 and a property ϕ expressed in Linear-time Temporal Logic (LTL)[23] and/or Metric Temporal Logic (MTL) [3]. Using *S*BIP, we can perform two types of analysis:

1. Quantitative: we estimate the probability that the system `S` satisfies a given property ϕ.
2. Qualitative: we test whether the probability of a given property ϕ being satisfied by the system `S` is greater or equal to a certain threshold θ.

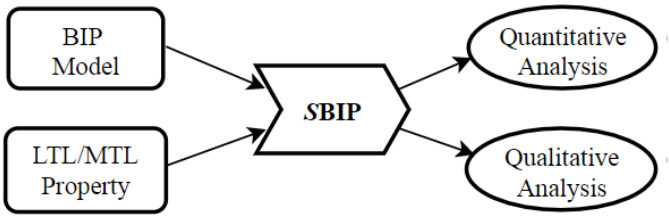

Fig. 5. *S*BIP Statistical Model Checker

To decide whether `S` satisfies ϕ (written $\texttt{S} \models \phi$), *S*BIP refers to simulation based techniques: Probability Estimation (PE) [13] for quantitative properties and Hypothesis Testing (HT) [29] for qualitative properties.

[2] http://www-verimag.imag.fr/BIP-SMC-A-Statistical-Model-Checking.html?lang=en

3.1 Quantitative Analysis

In this work, we use a stochastic bounded variant of LTL to express properties. In LTL, path formulas are defined using four bounded temporal operators namely, `Next` ($N\psi_1$), `Until` ($\psi_1 \cup^k \psi_2$), `Eventually` ($F^k\psi_1$), and `Always` ($G^k\psi_1$), where k is an integer value that specifies the length of the considered system execution trace and ψ_1,ψ_2 are called state formulas, which is a Boolean predicate evaluated on the system states.

*S*BIP allows to check parametric property ϕ(`x`), where `x` is a parameter ranging over a finite instantiation domain. It also provides a summary of analysis results and generates specific curves and/or plots of results. We present four examples of quantitative properties:

Property 1: the probability of water height levels on April 27.
In LTL: $P_{=?}[F^{1000}\ (WH\_L = L\ \&\&\ Day = 117)]; \quad L = 1:5:1;$

The results are given in Figure 6. We find that level 5 is the most likely and levels 4 and 3 are less likely. However, levels 1 and 2 are never observed on this day. These predictions concerning water height sensor and estimations from other sensors can help the managers of dam infrastructure to adjust the spillgate level.

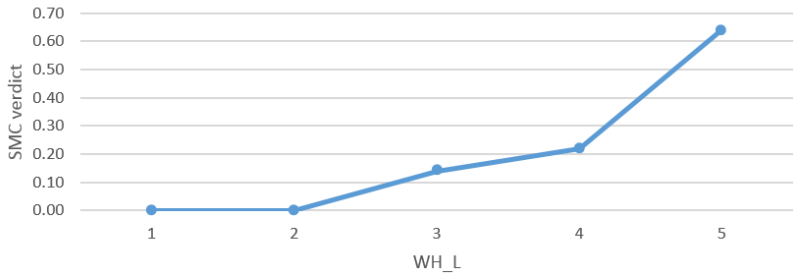

Fig. 6. Probability of water height levels on April 27

Property 2: the probability of each level of water height at the first weeks of January and May.
In LTL:

$$\begin{cases} P_{=?}[F^{1000}\ (WH\_L = L\ \&\&\ Day = T)]; \quad T = 1:7:1; T = 121:127:1; \\ \quad L = \{1,2,3,4,5\} \end{cases}$$

Figure 7 shows the SMC verdict of property 2. We see that level 5 is rarely observed in the first week of January, however, this level is most likely in the first week of May. The opposite for levels 1 and 2, which are more possible in the first week of January and rare in the first week of May. With LTL properties, we can predict the evolution of water height level at any period of the year.

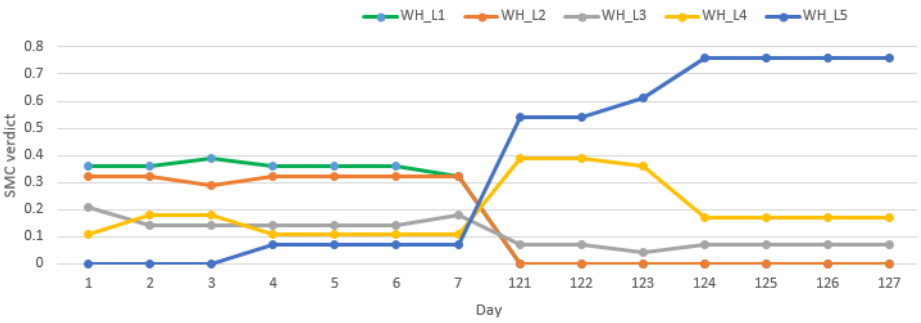

Fig. 7. Probability of water height levels at first weeks of January and May

Property 3: the probability that the water height level stays on the same level the last week of May.

In LTL:

$$\begin{cases} P_{=?}[G^{1000}\ (WH\_L = L\ \&\&\ Day = 145) \cup^{1000}\ (WH\_L = L\ \&\&\ \ Day = T)]; \\ T = 146 : 151 : 1; \quad L = \{1, 2, 3, 4, 5\} \end{cases}$$

As shown in Figure 8, there is a high possibility that the water height level will remain at levels 4 or 5 in the last week of May.

Property 4: the probability that the water height changes from first level on January 16th to other levels on the next day.

In LTL:

$$\begin{cases} P_{=?}[\ (WH\_L = 1\ \&\&\ Day = 16) \cup^{1000}\ (WH\_L = L\ \&\&\ \ Day = 17)\ ]; \\ L = \{2, 3, 4, 5\} \end{cases}$$

Figure 9 shows that change to levels 2 and 3 is most likely while there is little chance of change to levels 4 and 5.

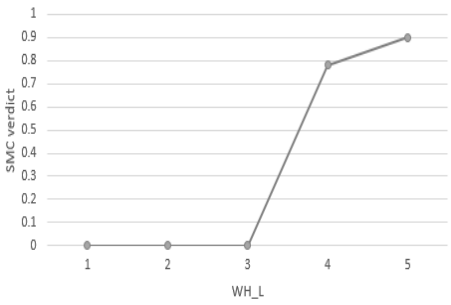

Fig. 8. Results of property 3

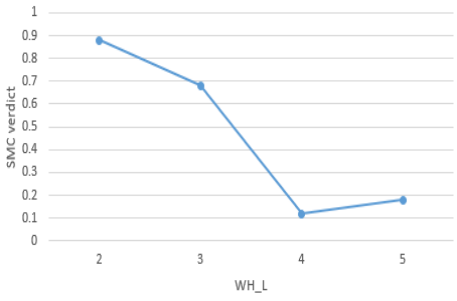

Fig. 9. Results of property 4

3.2 Qualitative Analysis

For qualitative analysis of sensor behavior, we rate sensors' readings based on their probabilities as following:

1. Not observed (RED): never seen in 28 years.
2. Rare (ORANGE): observed once or twice within 28 years.
3. Possible (YELLOW): observed 3 to 21 times in 28 years.
4. Very possible (GREEN): observed more than 21 times.

Table 1 defines the possible probabilities. Based on these considerations, we express qualitative properties that allow testing the compliant of sensors' readings with the learned model.

State	Not observed	Rare	Possible	Very Possible
Probability	0	]0, 0.09]	]0.09, 0.75]	]0.75, 1]

Table 1. Sensor State Rate

Property 5: Check whether the probability that water height reaches level 5 is higher than 0.75.

In LTL: $P_{>0.75}[F^{1000}\ (WH\_L = 5\ \&\&\ Day = T)]; \quad T = 1 : 365 : 1;$

Figure 10 shows the results provided by *S*BIP. This property allows calculating the set $DL5_{vp} = \{124, .., 202\}$ of days where the level 5 of water height is very possible.

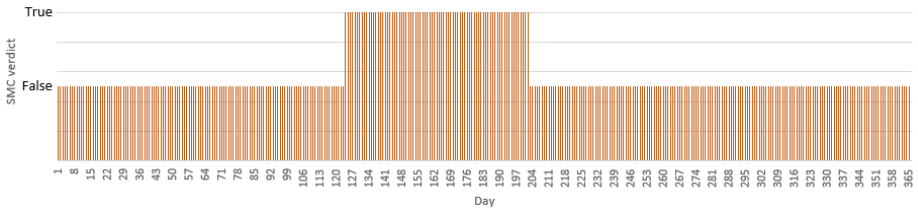

Fig. 10. Probability that water height level 5 is very possible

In the same way, we can calculate the sets $DL4_{vp}$, $DL3_{vp}$, $DL2_{vp}$, $DL1_{vp}$ where levels 4, 3, 2, and 1 are very possible. Based on these calculations, we define the function $isVeryPossibe$ as:

$$isVeryPossibe(WH\_L, Day) \leftarrow$$
$$(WH\_L = 5 \;\&\&\; Day \in\; DL5_{vp} \;||\; WH\_L = 4 \;\&\&\; Day \in\; DL4_{vp} ||$$
$$WH\_L = 3 \;\&\&\; Day \in DL3_{vp} \;||\; WH\_L = 2 \;\&\&\; Day \in\; DL2_{vp} \;||$$
$$WH\_L = 1 \;\&\&\; Day \in\; DL1_{vp})$$

We have also defined the functions $isPossible$, $isRare$, and $isNotObserved$ which allow respectively to check if the data collected by the sensors are possible, rarely observed, or never observed.

The defined functions are used to build the model of Figure 11 that allows evaluating the conformity of any water height sensor reading regarding the provided trace. The model can help to distinguish between anomalous and correct sensor readings.

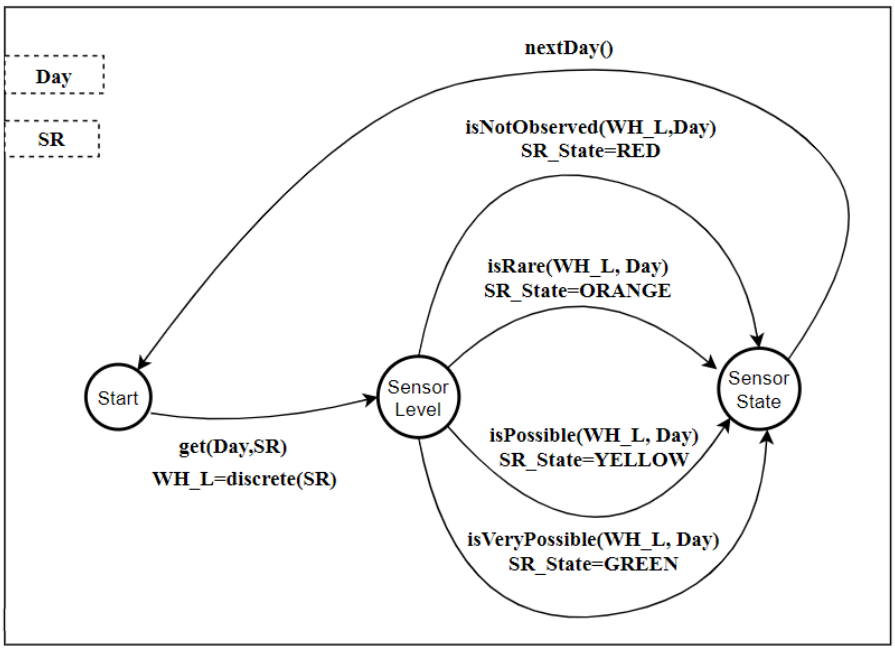

Fig. 11. Sensor State Model

The sensor state model can be used to check the quality of sensed data from the existing trace. In Figure 12, we discover very possible readings (Green points), possible readings (Yellow points), and rare readings (Orange points) in the months April and May of 2016. As shown in the Figure, some rare readings are detected at the beginning of April and May.

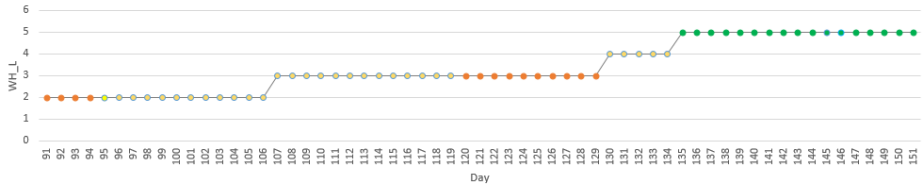

Fig. 12. Score of water height sensor data for April and May of 2016

The sensor state model also allows for checking new observations. Figure 13 presents the test results for April and May of 2017. We see that no unusual observation is found and that the observations of Avril are possible and the observations of May are highly possible.

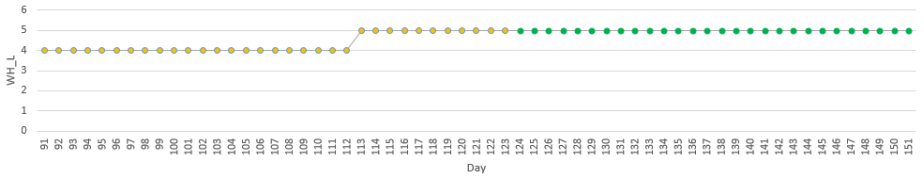

Fig. 13. Score of water height sensor data for April and May of 2017

4 Related Work

Time series analysis is one of the active areas of research due to its application in different fields, such as in the context of IoT-based systems. For sensors time series data, predicting the next measurements and detecting erroneous readings are the relevant tasks. The paper [11] presents the several approaches proposed for this purpose:

- Statistical approaches such as the method proposed by [30] that builds a window-based forecasting model from past observations, then it classifies the sensors' readings as anomalous based on a given prediction confidence interval.
- Probabilistic approaches use probabilistic models such as Bayesian Networks (BNs) [14] to measure the probability of sensors' readings. However, these approaches do not scale well.
- Proximity-based or clustering-based approaches such as [12, 6] use distances between the sensed data to detect the erroneous readings. For high dimensional data, these approaches do not work well.

– Prediction-based approaches such as [25, 16] use machine learning methods to predict the sensors' readings based on a model trained from past observations. However, training is time-intensive.

In our approach, we generate stochastic automata that specify the sensor behavior from past observations, and then we apply SMC to simulate the learned model and express LTL properties that predict the sensors' readings and analyze the sensor behavior in time. SMC is a powerful technique that handles scalability and requires less memory and time. The paper [1] provides a survey of the existing SMC tools. *S*BIP tool used in this work was applied for the analysis of various systems [21, 20, 5]. Our approach is different from all the approaches presented above. It allows to build a behavioral automata-based model from data and analyze this model using formal verification techniques. Among the works in this direction:

– The authors in [24] use Extended Finite Automata and residuals techniques to detect deviations of the behavior of the inhabitant in a smart home from a log of binary sensor events.
– The paper [18] models logs from SCADA systems using timed automata and applies the UPPAAL model checker to express a set of logic properties for detecting attacks targeting these systems.
– [19] uses Markov Decision Process for modeling the behavior of elastic cloud applications based on past log and then introduces probabilistic model checking to perform cloud elasticity decision using PCTL properties.
– [10] specifies a stochastic model in Deterministic-Time Markov Chain from the architecture description of the managed system considering different metrics related to cloud-infrastructure execution traces. Then, the PRISM model checker is used to optimize the self-adaptation decisions.

5 Conclusion

We presented an approach for a formal analysis of sensors' behavior. A formal model expressed as stochastic automata has been derived from sensor time series data then quantitative LTL properties expressed on this model are used to predict sensor readings. Also, qualitative LTL properties are used for defining a second automata-based model that allows checking if the new measurements are compliant with past observations. We have applied our approach to analyzing the behavior of three sensors from a dam infrastructure at different times. Our approach provides several advantages, including:

– We use BIP formalisms that allow the rigorous specification and analysis of sensor behavior.
– We use a component-based approach supported by BIP that facilitates portraying sensors behavior with reusability, and maintainability features.
– We developed a prototype that automatically generates sensor behavior and sensor state models from any existing traces.

– We use statistical model checkers that consume less memory and can check models with large state spaces.

In the future, we are planning to enhance the proposed approach by analyzing the consistency between the behaviors of a set of sensors and expressing inter-sensors properties.

6 Acknowledgments

The research leading to these results has been supported by the European Union through the BRAIN-IoT project H2020-EU.2.1.1. Grant agreement ID: 780089. The authors would like to thank EMALCSA Company for the data collected from the dam infrastructure.

References

1. Agha, G., Palmskog, K.: A Survey of Statistical Model Checking. ACM Transactions on Modeling and Computer Simulation **28**(1), 1–39 (Jan 2018). https://doi.org/10.1145/3158668
2. Al-Turjman, F., Malekloo, A.: Smart parking in IoT-enabled cities: A survey. Sustainable Cities and Society **49**, 101608 (2019)
3. Alur, R., Henzinger, T.: Real-Time Logics: Complexity and Expressiveness. Information and Computation **104**(1), 35–77 (May 1993). https://doi.org/10.1006/inco.1993.1025
4. Basu, A., Bensalem, S., Bozga, M., Combaz, J., Jaber, M., Nguyen, T.H., Sifakis, J.: Rigorous Component-Based System Design Using the BIP Framework. IEEE Software **28**(3), 41–48 (May 2011)
5. Beaulaton, D., Said, N.B., Cristescu, I., Sadou, S.: Security Analysis of IoT Systems Using Attack Trees. In: Albanese, M., Horne, R., Probst, C.W. (eds.) Graphical Models for Security, vol. 11720, pp. 68–94. Springer International Publishing, Cham (2019)
6. Breunig, M.M., Kriegel, H.P., Ng, R.T., Sander, J.: LOF: identifying density-based local outliers. ACM SIGMOD Record **29**(2), 93–104 (Jun 2000). https://doi.org/10.1145/335191.335388
7. Daissaoui, A., Boulmakoul, A., Karim, L., Lbath, A.: IoT and Big Data Analytics for Smart Buildings: A Survey. Procedia Computer Science **170**, 161 – 168 (2020). https://doi.org/https://doi.org/10.1016/j.procs.2020.03.021
8. David, A., Larsen, K.G., Legay, A., Mikučionis, M., Poulsen, D.B.: Uppaal SMC tutorial. International Journal on Software Tools for Technology Transfer **17**(4), 397–415 (Aug 2015)
9. Dougherty, J., Kohavi, R., Sahami, M.: Supervised and unsupervised discretization of continuous features. In: Prieditis, A., Russell, S. (eds.) Machine Learning Proceedings 1995, pp. 194 – 202. Morgan Kaufmann, San Francisco (CA) (1995). https://doi.org/https://doi.org/10.1016/B978-1-55860-377-6.50032-3
10. Franco, J.M., Correia, F., Barbosa, R., Zenha-Rela, M., Schmerl, B., Garlan, D.: Improving self-adaptation planning through software architecture-based stochastic modeling. Journal of Systems and Software **115**, 42–60 (May 2016). https://doi.org/10.1016/j.jss.2016.01.026

11. Giannoni, F., Mancini, M., Marinelli, F.: Anomaly Detection Models for IoT Time Series Data. ArXiv **abs/1812.00890** (2018)
12. He, Z., Xu, X., Deng, S.: Discovering cluster-based local outliers. Pattern Recognition Letters **24**(9-10), 1641–1650 (Jun 2003). https://doi.org/10.1016/S0167-8655(03)00003-5
13. Hérault, T., Lassaigne, R., Magniette, F., Peyronnet, S.: Approximate probabilistic model checking. In: Verification, Model Checking, and Abstract Interpretation. pp. 73–84. Springer Berlin Heidelberg, Berlin, Heidelberg (2004)
14. Hill, D.J., Minsker, B.S., Amir, E.: Real-time Bayesian anomaly detection in streaming environmental data: REAL-TIME BAYESIAN ANOMALY DETECTION. Water Resources Research **45**(4) (Apr 2009). https://doi.org/10.1029/2008WR006956
15. Kwiatkowska, M., Norman, G., Parker, D.: Prism 4.0: Verification of probabilistic real-time systems. In: Gopalakrishnan, G., Qadeer, S. (eds.) Computer Aided Verification. pp. 585–591. Springer Berlin Heidelberg, Berlin, Heidelberg (2011)
16. Malhotra, P., Vig, L., Shroff, G., Agarwal, P.: Long short term memory networks for anomaly detection in time series. In: ESANN (2015)
17. Mediouni, B.L., Nouri, A., Bozga, M., Dellabani, M., Legay, A., Bensalem, S.: SBIP 2.0: Statistical Model Checking Stochastic Real-time Systems. In: ATVA 2018 - 16th International Symposium Automated Technology for Verification and Analysis. pp. 536–542. Springer, Los Angeles, CA, United States (Oct 2018). https://doi.org/10.1007/978-3-030-01090-4_33
18. Mercaldo, F., Martinelli, F., Santone, A.: Real-Time SCADA Attack Detection by Means of Formal Methods. In: 2019 IEEE 28th International Conference on Enabling Technologies: Infrastructure for Collaborative Enterprises (WETICE). pp. 231–236. IEEE, Napoli, Italy (Jun 2019). https://doi.org/10.1109/WETICE.2019.00057
19. Naskos, A., Gounaris, A., Mouratidis, H., Katsaros, P.: Online Analysis of Security Risks in Elastic Cloud Applications. IEEE Cloud Computing **3**(5), 26–33 (Sep 2016). https://doi.org/10.1109/MCC.2016.108
20. Nouri, A., Bensalem, S., Bozga, M., Delahaye, B., Jegourel, C., Legay, A.: Statistical model checking QoS properties of systems with SBIP. International Journal on Software Tools for Technology Transfer **17**(2), 171–185 (Apr 2015)
21. Nouri, A., Mediouni, B.L., Bozga, M., Combaz, J., Bensalem, S., Legay, A.: Performance evaluation of stochastic real-time systems with the SBIP framework. International Journal of Critical Computer-Based Systems **8**(3/4), 340 (2018)
22. Park, C., Kim, Y., Jeong, M.: Influencing factors on risk perception of IoT-based home energy management services. Telematics and Informatics **35**(8), 2355 – 2365 (2018)
23. Pnueli, A.: The temporal logic of programs. In: 18th Annual Symposium on Foundations of Computer Science. pp. 46–57. IEEE Computer Society, USA (oct 1977). https://doi.org/10.1109/SFCS.1977.32
24. Saives, J., Pianon, C., Faraut, G.: Activity Discovery and Detection of Behavioral Deviations of an Inhabitant From Binary Sensors. IEEE Transactions on Automation Science and Engineering **12**(4), 1211–1224 (Oct 2015). https://doi.org/10.1109/TASE.2015.2471842
25. Shahid, N., Naqvi, I.H., Qaisar, S.B.: One-class support vector machines: analysis of outlier detection for wireless sensor networks in harsh environments. Artificial Intelligence Review **43**(4), 515–563 (Apr 2015). https://doi.org/10.1007/s10462-013-9395-x

26. Tao, Z.: Advanced Wavelet Sampling Algorithm for IoT based environmental monitoring and management. Computer Communications **150**, 547 – 555 (2020). https://doi.org/https://doi.org/10.1016/j.comcom.2019.12.006
27. Yang, Y., Webb, G.I., Wu, X.: Discretization Methods, pp. 101–116. Springer US, Boston, MA (2010)
28. Yi Xie, Shun-Zheng Yu: A Large-Scale Hidden Semi-Markov Model for Anomaly Detection on User Browsing Behaviors. IEEE/ACM Transactions on Networking **17**(1), 54–65 (Feb 2009). https://doi.org/10.1109/TNET.2008.923716
29. Younes, H.L.S., Simmons, R.G.: Probabilistic verification of discrete event systems using acceptance sampling. In: Brinksma, E., Larsen, K.G. (eds.) Computer Aided Verification. pp. 223–235. Springer Berlin Heidelberg (2002)
30. Yu, Y., Zhu, Y., Li, S., Wan, D.: Time Series Outlier Detection Based on Sliding Window Prediction. Mathematical Problems in Engineering **2014**, 1–14 (2014). https://doi.org/10.1155/2014/879736