Probabilistic Model Checking for IEC 61499: A Manufacturing Application

Irman Faqrizal[‡], Tatiana Liakh[*], Midhun Xavier[*], Gwen Salaün[‡], Valeriy Vyatkin[*†]
[*]Department of Computer Science, Computer and Space Engineering, Lulea Tekniska Universitet, Sweden
[†]Department of Electrical Engineering and Automation, Aalto University, Espoo, Finland
[‡]Univ. Grenoble Alpes, CNRS, Grenoble INP, Inria, LIG, Grenoble, France

Email: {irman.faqrizal, gwen.salaun}@inria.fr, {tatiana.liakh, midhun.xavier}@ltu.se, vyatkin@ieee.org

***Abstract*—The ever-increasing complexity of industrial control systems generates a demand for reliable development methods. IEC 61499, a recent industrial standard, helps to develop complex distributed systems based on their positive characteristics, namely reusability, reconfigurability, interoperability, and portability. Formal verification techniques, such as model checking, have been proposed to ensure the correctness of these systems during the design time. However, they do not consider the presence of the environment that can impact the application behaviour at runtime. This work combines design time and runtime analyses to apply probabilistic model checking on an IEC-61499-based manufacturing application. We present several probabilistic properties to be checked. The results are visualised graphically to be analysed, which allows one to optimise the system's quantitative features, such as productivity.**

***Index Terms*—Industrial control systems, IEC 61499, formal verification, probabilistic model checking.**

I. Introduction

An industrial control system (ICS) regulates the physical processes of a system in an industrial environment to achieve specific objectives [1]. One of the leading trends in ICS is the flexibility and reconfigurability of production lines. Recent projects, such as [2], focus on developing technologies that make it easy to adapt production lines to constantly changing market requirements and seamlessly integrate devices from different vendors. Reconfiguration of the production line and independent actors (e.g. AGVs or employees) can have a tangible impact on the overall productivity.

IEC 61499 [3] is a programming standard for developing ICS software, known for its reusability, reconfigurability, interoperability, and portability [4]. It was developed as an extension of the IEC 61131-3 standard [5] for distributed systems. IEC 61499 defines the system behaviour using interconnected logical components called function blocks. A large distributed industrial system can be created using different control devices, and its control software may be composed of many function blocks, but this complexity can make it prone to errors. Also, the cost of errors is high in such systems. Formal verification techniques, such as model checking [6], have been proposed in [7]–[10] to ascertain the correctness of IEC 61499-based systems. However, they do not consider the presence of the environment, which includes unpredictable external aspects influencing the system's execution.

It is crucial to take into account the environment when verifying the system's quantitative aspects, such as productivity and energy consumption. An example of an environment is the workers operating on the system. A worker may interact with the system in a certain way such that events associated with energy consumption occur more frequently than expected. In such a case, conventional model checking is insufficient since it does not deal with the likelihood of the event's occurrence.

Probabilistic model checking (PMC) is a formal method to verify quantitative properties of stochastic systems [11]. It represents a system as a mathematical model enriched with the probability of executing sequences of events. PMC helps to verify control systems' behaviour when considering the environment's presence. It allows us to first infer the probabilistic values in the model by monitoring the system execution, which is influenced by the environment. Then, we can verify the probability of sequences of events to occur, formulated as probabilistic properties, according to the enriched model.

The main contributions of this paper are (i) the application of PMC on an IEC-61499-based manufacturing system and (ii) the analysis of its results for optimising the system's quantitative aspect, which in our case is productivity. The industrial case study is a drilling station, which is a manufacturing application that drills materials on a rotating table. The environments are represented as strategies for interacting with the materials on the table. We rely on the techniques described in [12], which combines static and dynamic analyses, to generate the probabilistic model of the system. Several probabilistic properties are proposed to be checked on the generated model. The results are displayed graphically to observe the impact of different strategies on the drilling station. We show that analysing these results allows us to choose which strategy can optimise productivity and to propose improved strategies.

The rest of the paper is structured as follows. Section II introduces background notions. Section III explains PMC for IEC 61499. Section IV describes the application of PMC. Section V surveys related work. Section VI concludes.

II. Background

There exist standards for developing control systems, such as IEC 61131-3 [5] and IEC 61499 [3]. The former is a conventional standard for developing ICSs using ladder and

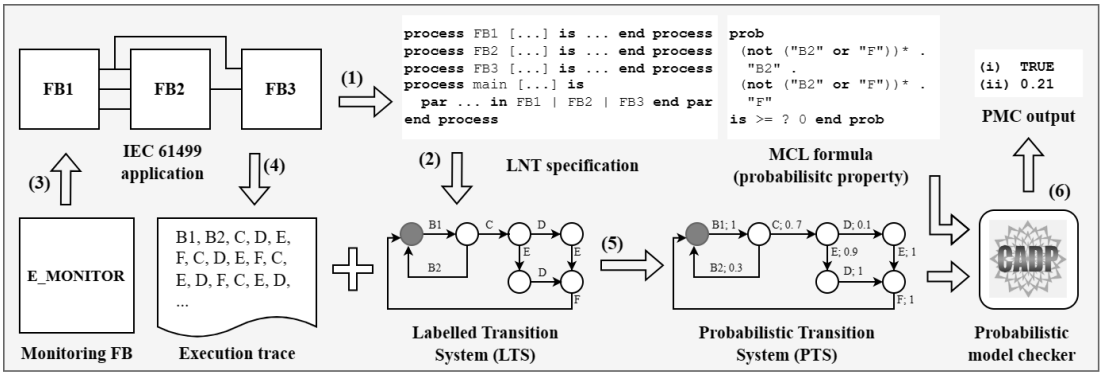


Fig. 1: Probabilistic model checking for IEC 61499

function block diagrams, structured text, and sequential function charts. It adopts a scan-based model where the CPU scans what will be executed after a certain period. We focus on the latter, which relies on an event-driven execution model and supports distributed design by assigning a single application to multiple control devices.

IEC 61499 defines the notion of function block (FB). There are three types of FB: basic, service interface, and composite. An FB encapsulates an internal behaviour depending on its type. Basic FBs describe their behaviours using execution control charts. Service interface FBs are programmed depending on the respective control devices. Composite FBs are composed of networks of FBs. An IEC 61499 application is composed of FBs that are connected through their event and data interfaces (as illustrated in Fig. 1).

Model checking [6] is a method that verifies whether a system model satisfies a given property. The model checker returns a counterexample, a sequence of actions leading to property violation when the property is unsatisfied. This method can assert properties such as something bad never happens (i.e., safety) and something good eventually happens (i.e., liveness). On the other hand, probabilistic model checking (PMC) can verify probabilistic properties on a model enriched with the likelihood of executing sequences of events.

In this work, we use the CADP [13] model checker. A system is specified in the LNT [14] specification language, an extension of LOTOS [15], an ISO-standardized process algebra. A property is written as MCL [16] formula, a temporal logic suitable for expressing concurrent system properties. A behavioural model called labelled transition system (LTS) is generated from the LNT specification. An LTS is a state machine with states and transitions, where each transition is labelled with an event. CADP supports PMC by extending the LTS into a probabilistic model called probabilistic transition system (PTS). A transition in PTS is labelled by an event and the probability of executing it.

The approach presented in this paper is generic. Instead of PTS, we may use discrete-time Markov chain (DTMC) described in [11] as the probabilistic model. Other tools that support PMC are also applicable. For instance, the PRISM [17] model checker can be an alternative to CADP.

III. PMC FOR IEC 61499

Conventional model checking can verify that an IEC 61499 application triggers a sequence of events. The model checker returns whether this is true or false. PMC allows us to go beyond this by returning the probability of triggering that sequence. However, there are more steps besides modelling and formulating the property. The application must be monitored to obtain execution traces, which are sequences of events triggered by the application. A probabilistic model is computed from the initial model and the traces. PMC for IEC 61499 is illustrated in Fig. 1 and summarised as follows:

(1) The application is translated into an LNT specification.
(2) The specification is compiled into an LTS model.
(3) A monitoring FB is integrated into the application.
(4) Traces are obtained when the application is executed.
(5) A PTS is computed from the LTS and the traces.
(6) CADP checks a probabilistic property, expressed as an MCL formula, on the PTS.

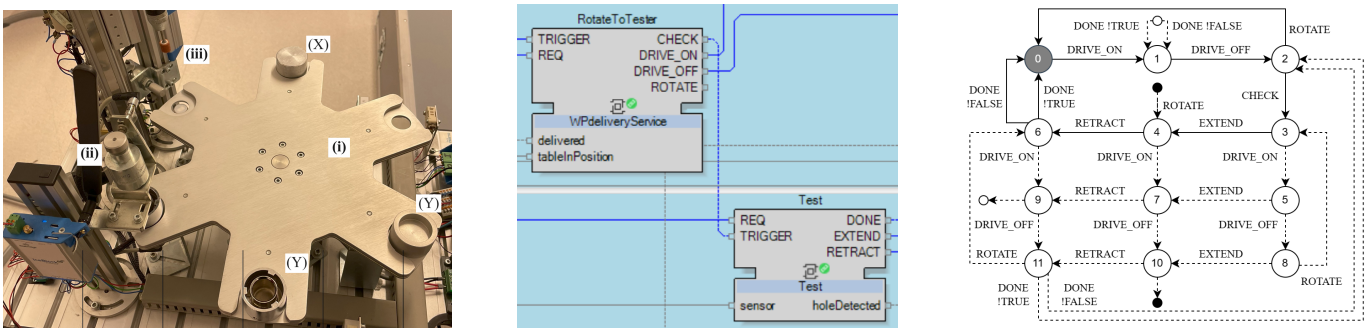
The translation from an IEC 61499 application to LNT specification is done by constructing a subprocess for each FB. Then, a parallel composition consisting of these processes is built as the main process. CADP is used to obtain an LTS from the translated LNT specification. The monitoring FB consists of input interfaces connected to the existing FBs' output interfaces. An event is added to the execution trace whenever one of the event interfaces in the application is triggered. A PTS is generated by traversing the LTS using the traces. When the PTS is checked according to the probabilistic property, CADP returns (i) a verdict and (ii) the probability of executing the sequences specified in the property. In this work, we focus on the second output. Several properties are checked on the same system with different environments to obtain several probabilities for analysis purposes.

More details of the approach, such as the translation patterns from IEC 61499 to LNT and the algorithm for computing PTS, are given in [12].

IV. APPLYING PMC TO A MANUFACTURING APPLICATION

A. Drilling station

As shown in Fig. 2a, the drilling station system consists of several mechatronic components: (i) a table, (ii) a tester,



(a) Drilling station (b) `RotateToTester` and `Test` (c) Behavioural model (LTS)

Fig. 2: Description and model of drilling station

and (iii) a drill. These mechatronic components are considered smart, i.e., equipped with their own control devices that implement basic operations. The table component undergoes rotation from one fixed position to another. A complete cycle is achieved when the table rotates six times. Whenever a material is in the loading position, the table rotates to align it under the tester component. The tester then checks that the material has been drilled. If it has already been drilled, the drill component is not activated. Otherwise, the drill component is activated to drill the material as soon as its sensor detects the presence of the material beneath it. Materials that have not been drilled are called solid materials (X), while those that have been drilled are called void materials (Y).

The IEC 61499 application for the drilling station consists of 21 FBs. However, we focus on `RotateToTester` and `Test` FBs depicted in Fig. 2b. They are chosen because they correspond to the detection of materials on the table, which is the nondeterministic external aspect of the system. It is not necessary to consider other FBs that have deterministic behaviour with respect to the environment. For example, when there is solid material under the drill component, it is certain that the drilling process, described by the respective FBs, will start. Moreover, considering only essential FBs also allows us to minimise the model's size.

In `RotateToTester`, DRIVE_ON and DRIVE_OFF trigger events to start and stop the table rotation, respectively. CHECK is triggered when there is a material; otherwise, ROTATE is triggered. In `Test`, EXTEND and RETRACT are triggered sequentially to check the material type. DONE is triggered with holeDetected equals false when it is a solid material; if the material is void, holeDetected is true.

B. Formal model

As discussed in Section III, the first step is to model the application. `RotateToTester` and `Test` FBs are translated into LNT processes (not shown) to generate an LTS presented in Fig. 2c. We focus on the output event interfaces. The boolean value preceded by an exclamation mark is the value of the associated data interface (e.g., in transitions from states 6 to 0, the event DONE is associated with the data interface holeDetected). An example of a sequence of events representing a rotation is DRIVE_ON, DRIVE_OFF, ROTATE. The table rotated once in this case, and no material was detected.

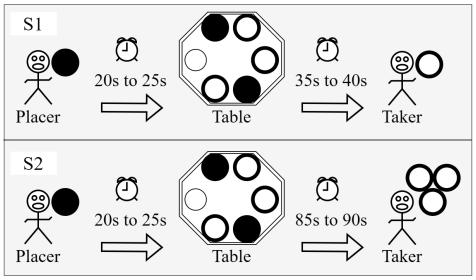


Fig. 3: Strategies for placing and taking the materials

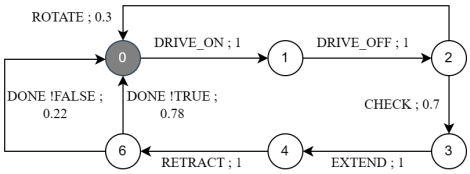
C. Experiment

This experiment aims to show that PMC is helpful in analysing the impact of the system environment for optimising the system's productivity. For this, we define a scenario where the environment is represented as different strategies to place and take materials on the drilling station table. Then, we explain how the system is executed to obtain the execution traces for generating the probabilistic models.

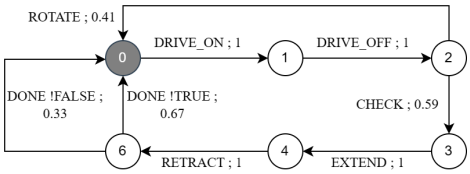
Scenario. Two workers are involved in the drilling station. *Placer* puts solid materials on the table, while *Taker* picks the void ones. They have other responsibilities because the drilling station is part of a larger industrial system. Therefore, they are not always stationed at their location. The notion of environment varies depending on the system. In more complex systems, the role of Placer and Taker can be replaced by other external aspects (e.g., room temperature) that impact the system in a more complicated manner.

Here, the strategy that Placer and Taker use represents the environment. In this experiment, two strategies illustrated in Fig. 3 are chosen. The void materials are taken individually in the first strategy, whereas the second one takes all of them at once but after a longer period of time. These are chosen because it is not easy to find the best strategy.

(S1) Placer places a solid material every 20 to 25 seconds, and Taker picks a void material every 35 to 40 seconds.



(a) PTS computed from traces of S1



(b) PTS computed from traces of S2

Fig. 4: PTSs

(S2) Placer places a solid material every 20 to 25 seconds, and Taker picks all void materials every 85 to 90 seconds.

System execution. The system is executed and monitored by a monitoring FB for 10 minutes using the aforementioned strategies. Two execution traces containing nearly 700 events are obtained. S1 is applied in the first five minutes to get the first trace, and for the second half, we use S2. These traces, along with the LTS in Fig. 2c, are used to generate two PTSs shown in Fig. 4. The transitions are now enriched with the probabilities of triggering the events. For instance, in state 2 of the PTS in Fig. 4a, after DRIVE_OFF is triggered, there is a 30% probability to trigger ROTATE. Furthermore, as shown in both PTSs, the dashed transitions in the initial LTS (Fig. 2c) are hidden because the probability of executing them is 0%. In addition to these two PTSs, we use simplified *sliding window* [18] techniques in which multiple PTSs are generated for every n events using the last m events.

Observing the PTS allows us to see the probability of event sequences. However, techniques such as PMC are required for more specific results that are not straightforward to obtain manually. For instance, it is not straightforward to compute the probability of detecting three void materials in six rotations (i.e., a complete cycle) only by looking at the PTS. Moreover, a more complex model may contain millions of states. Hence, it would be impossible to analyse the PTS visually.

D. Probabilistic properties

The drilling station uses the drill component to transform solid materials into void materials. Therefore, the system should drill as many materials as possible during its execution to optimise productivity. We focus on the following probabilistic properties to measure the productivity of the system.

(P1) the probability of detecting materials,
(P2) the probability of having certain numbers of materials in the system,
(P3) and the probability of detecting a solid material in several rotations.

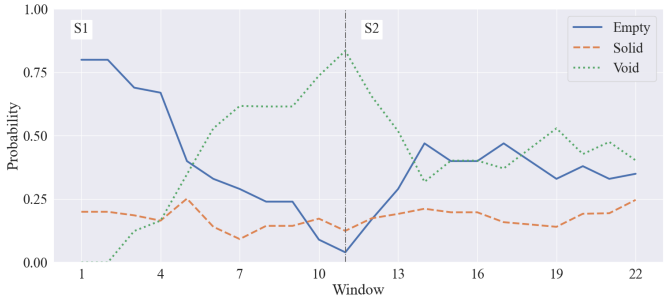


Fig. 5: Probability of detecting materials

P1 helps to analyse system productivity by showing how often a type of material is detected during system execution. For example, when the probability of detecting a solid material is high, it means that the drilling component is used frequently. We check this property as the system is running to provide dynamic information to the user. In particular, the workers involved in the system may use this information to improve productivity at runtime.

P2 and P3 are checked after the drilling station has finished running. In P2, the number of materials in the system means the number of materials after a complete cycle (i.e., the table rotated six times). The results can be used as a summary for evaluation purposes. For example, we can check the probability of having six void materials on the table. The high probability indicates that the worker in charge did not take the materials as often as they should have during the system execution. P3 focuses on the detection of solid material. It informs the user about the probability of detecting a solid material after some rotations. It can also show how many rotations are required to make the probability of detecting a solid material close to 100%.

Several MCL formulas are written for each property. In P1, there are three distinct formulas for three types of detection: detection of solid material, void material, and empty place. For example, the following MCL formula returns the probability of detecting solid materials.

```
prob (not ("CHECK" or "ROTATE"))* .
     "CHECK" .
     (not ("DONE !TRUE" or "DONE !FALSE"))* .
     "DONE !FALSE"
is >= ? 0 end prob
```

The part between **prob** and **is** in the formula is essentially a regular expression corresponding to a sequence of events. The model checker returns the probability of that sequence according to the probabilistic model. In the above formula, the sequence consists of CHECK followed by DONE !FALSE. CHECK means a material is detected, whereas the second event means that it is a solid material.

E. Results and Discussion

Results. Fig. 5 shows the probability of detecting materials (P1). Window (x-axis) represents the PTSs generated over time using the sliding window techniques. In this case, a PTS is generated every 30 events using the last 100 events to obtain

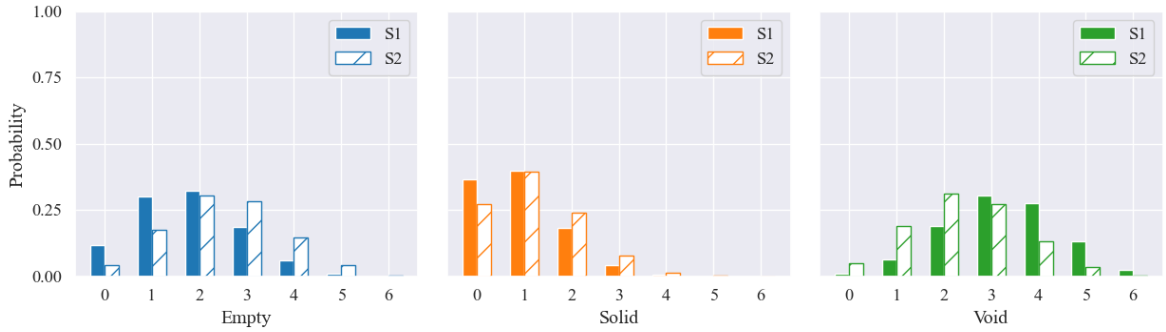


Fig. 6: Probability of the numbers of materials in the system

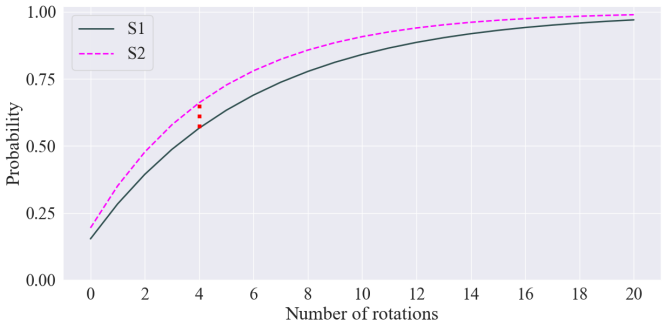


Fig. 7: Probability of detecting a solid material in several rotations

an observable curve chart. The probability of detecting void materials increases up to 77% when S1 is applied. It decreases and remains close to 50% in the second half when the strategy is changed to S2. In contrast, the probability of empty places gets lower when using S1 until reaching 5%; then, it returns to about 45% when S2 is used. The probability of solid material detection stays just under 25% in both strategies.

Fig. 6 shows the probability of having certain numbers of materials in the system (P2). The probability of empty places is overall higher using S2 compared to S1; with S2, there is 5% probability that there are five empty places. The probability of having solid materials on the table is also higher when S2 is used, with the probability of having two of them close to 25%. The probability of void materials in the system is higher using S1. There is a 4% probability that void materials fully occupy the table.

Fig. 7 shows the probability of detecting a solid material in several rotations (P3). S2 requires fewer rotations to get a higher probability. As highlighted, the peak difference of probabilities between the two strategies occurs in four rotations (57% and 66%, respectively, for S1 and S2). Note that 13 solid materials were processed into void materials using the first strategy, whereas the second strategy drilled 15 materials.

Discussion. The results show that the two strategies representing two environments have different impacts on the drilling station. There are two important points from this experiment. Firstly, S2 is better than S1 for optimising productivity because the probability of detecting void materials is lower, allowing solid materials to be placed on the table. Secondly, the probability of detecting empty places is also higher in S2. Therefore, Placer should put solid materials on the table more frequently. For instance, we may propose a new strategy where solid material is placed every 10 to 15 seconds. Overall, analysing the results of PMC can help to decide which environment (i.e., strategy) is better for optimising the quantitative aspects of the system (i.e., productivity). Furthermore, the results can help to determine the degree of component *wear and tear*. When the probability of detecting solid materials is high, one must anticipate replacing the drilling component.

V. Related work

In [10], model checking is applied to verify the correctness of a turntable system, which resembles the drilling station. Three model checkers were tested: Spin [19], CADP [13], and Uppaal [20]. The system, written in χ language, is translated to the model checkers' specification languages. Properties such as deadlock freedom and fairness were checked. This work and other notable ones in [7]–[9] focus on ensuring correctness by considering the model obtained during the design time. They do not consider the presence of the environment, such as workers interacting with the system. On the contrary, our approach considers the environment using probabilistic models computed from the execution traces.

In [21], PMC is applied to business process model and notation (BPMN) processes. A BPMN process is translated to a specification language and then to a behavioural model. An executable process generates execution traces for computing a probabilistic model. The approach is illustrated using a hardware retailer case study. Several properties were checked, such as the probability of taking extra insurance when shipping a package. Their approach is comparable to ours because they also use the traces obtained from executing the system to enrich the model. However, they focus on modelling language for business processes (BPMN), while our work targets an industrial system development framework (IEC 61499).

The work in [12] combines static and dynamic analyses to generate probabilistic models of IEC 61499 applications. These models are then checked using PMC to verify several probabilistic properties. However, there is a lack of exper-

imental results. It does not show how PMC can be used to analyse the impact of the environment and infer possible improvements. In this paper, we apply the approach proposed in [12] to an industrial case study. An experiment is conducted to visualise graphically the results of PMC. We show that analysing these results is useful for optimising productivity.

The authors in [22] propose to add an enforcer FB to modify the behaviour of an IEC 61499 application according to the property. A property is formulated as a state machine in which the users can specify how to forward, discard, and replace events. The work is illustrated on a conveyor test station. The case study shows that the approach can allow the system to accept a material after two rejections in a row. This work aims to satisfy specific properties by adding an enforcer to the application. On the contrary, our technique adds a monitor to record the execution traces of the application. We take these traces as input to generate probabilistic models.

The works in [23]–[25] apply statistical model checking [26] (SMC) to analyse and optimise industrial systems in various aspects. In [23], SMC is combined with the fault tree analysis method to evaluate the probabilities of system failure and power consumption. The authors in [24] use system modelling language (SysML) to model cyber-physical systems. The case study on artificial pancreas shows that the approach can verify critical safety properties. Finally, SMC is used to analyse the performance of production lines in [25]. These works employ SMC, while ours uses PMC. SMC is known to have a lower memory requirement, but it returns approximative results as output. In contrast, PMC may require exploring the entire state space but provides more accurate results.

VI. Conclusion and future work

In this work, PMC is applied to analyse an IEC-61499-based system with respect to the environment's presence for optimising the system's quantitative aspects. We focus on a drilling station case study, where the environment is represented as the workers' (i.e., Placer and Taker) strategy when interacting with the system, and the objective is to optimise productivity. Several probabilistic properties were checked on the system's generated probabilistic models. We show that the results visualised using charts are helpful for exhaustive analysis of the impact of the environment on the system. This allows us to choose which strategy is better and propose new strategies to increase productivity.

The notion of environment in ICS varies depending on the system. It may include external aspects we have not considered, such as network communications between control devices (e.g., packet loss probability). Our main perspective for future work is to take these aspects into account. The PMC approach for IEC 61499 may need to be extended, depending on the environment we want to consider. For example, the model, traces, and property should be extended with timing constraints for network-related environments.

Acknowledgements. This work was supported by (i) the French National Research Agency in the framework of the « France 2030 » program (ANR-15-IDEX-0002), the LabEx PERSYVAL-Lab (ANR-11-LABX-0025-01), and (ii) the Horizon Europe Zero-SWARM project funded by the European Commission (grant agreement: 101057083).

References

[1] B. Galloway and G. P. Hancke, "Introduction to industrial control networks," *IEEE Comm. Surv. Tutor.*, vol. 15, no. 2, pp. 860–880, 2013.
[2] Zero-SWARM, "The Zero-SWARM project," https://zero-swarm.eu/.
[3] "International Electrotechnical Commission, Functional blocks - Part 1: Architecture, 2nd edn, IEC 61499-1," *IEC Geneva*, 2012.
[4] V. Vyatkin, "IEC 61499 as enabler of distributed and intelligent automation: State-of-the-art review," *IEEE Trans. Ind. Informatics*, vol. 7, no. 4, pp. 768–781, 2011.
[5] "Programmable controllers-part 3: Programming languages," *IEC 61131-3 (Ed. 2.0)*, 2002.
[6] C. Baier and J. Katoen, *Principles of model checking*. MIT Press, 2008.
[7] P. Ovsiannikova and V. Vyatkin, "Towards user-friendly model checking of IEC 61499 systems with counterexample explanation," in *Proc. of ETFA'21*. IEEE, 2021, pp. 1–4.
[8] D. Drozdov, S. Patil, V. Dubinin, and V. Vyatkin, "Formal verification of cyber-physical automation systems modelled with timed block diagrams," in *Proc. of ISIE'16*. IEEE, 2016, pp. 316–321.
[9] Z. E. Bhatti, P. S. Roop, and R. Sinha, "Unified functional safety assessment of industrial automation systems," *IEEE Trans. Ind. Informatics*, vol. 13, no. 1, pp. 17–26, 2017.
[10] E. M. Bortnik, N. Trcka, A. Wijs, B. Luttik, J. M. van de Mortel-Fronczak, J. C. M. Baeten, W. J. Fokkink, and J. E. Rooda, "Analyzing a *chi* model of a turntable system using SPIN, CADP and UPPAAL," *J. Log. Algebraic Methods Program.*, vol. 65, no. 2, pp. 51–104, 2005.
[11] M. Kwiatkowska, G. Norman, and D. Parker, *Probabilistic Model Checking: Advances and Applications*. Springer, 2018, pp. 73–121.
[12] Y. Falcone, I. Faqrizal, and G. Salaün, "Probabilistic analysis of industrial IoT applications," in *Proc. of IoT'22*. ACM, 2022, pp. 41–48.
[13] H. Garavel, F. Lang, R. Mateescu *et al.*, "CADP 2011: A Toolbox for the Construction and Analysis of Distributed Processes," *STTT*, vol. 15, no. 2, pp. 89–107, 2013.
[14] D. Champelovier, X. Clerc, H. Garavel, Y. Guerte, F. Lang, C. McKinty, V. Powazny, W. Serwe, and G. Smeding, "Reference Manual of the LNT to LOTOS Translator (Version 6.7)," 2018, INRIA/VASY and INRIA/CONVECS, 153 pages.
[15] "LOTOS — A Formal Description Technique Based on the Temporal Ordering of Observational Behaviour," ISO, Tech. Rep. 8807, 1989.
[16] CADP, "MCL manual page," https://cadp.inria.fr/man/mcl.html.
[17] M. Z. Kwiatkowska, G. Norman, and D. Parker, "PRISM: probabilistic symbolic model checker," in *Proc. of TOOLS'02*, ser. LNCS, vol. 2324. Springer, 2002, pp. 200–204.
[18] O. Muller, A. Butman, and M. Butman, "Opening a (sliding) window to advanced topics," in *Proc. of ITiCSE'17*. ACM, 2017, pp. 52–57.
[19] G. J. Holzmann, "The model checker SPIN," *IEEE Trans. Software Eng.*, vol. 23, no. 5, pp. 279–295, 1997.
[20] K. G. Larsen, P. Pettersson, and W. Yi, "UPPAAL in a nutshell," *Int. J. Softw. Tools Technol. Transf.*, vol. 1, no. 1-2, pp. 134–152, 1997.
[21] Y. Falcone, G. Salaün, and A. Zuo, "Probabilistic model checking of BPMN processes at runtime," in *Proc. of IFM'22*, ser. LNCS, vol. 13274. Springer, 2022, pp. 191–208.
[22] Y. Falcone, I. Faqrizal, and G. Salaün, "Runtime enforcement for IEC 61499 applications," in *Proc. of SEFM'22*, ser. LNCS, vol. 13550. Springer, 2022, pp. 352–368.
[23] A. Samadi, M. Ammar, and O. A. Mohamed, "Statistical model checking based analysis of fault trees and power consumption to enhance autonomous systems reliability," in *Proc. of NEWCAS'23*. IEEE, 2023, pp. 1–5.
[24] A. Alshalalfah, O. A. Mohamed, and S. Ouchani, "A framework for modeling and analyzing cyber-physical systems using statistical model checking," *Internet of Things*, vol. 22, p. 100732, 2023.
[25] P. Ballarini and A. Horváth, "Performance analysis of production lines through statistical model checking," in *Proc. of EPEW'21, and ASMTA'21*, ser. LNCS, vol. 13104. Springer, 2021, pp. 264–281.
[26] A. Legay, A. Lukina, L. Traonouez, J. Yang, S. A. Smolka, and R. Grosu, "Statistical model checking," in *Comput. and Softw. Science - State of the Art and Perspect.*, ser. LNCS. Springer, 2019, vol. 10000, pp. 478–504.