

CHAOS - Configurations Analysis of Swarms of Cyber-Physical Systems*

Valeria Trombetta^{1,*}, Maxime Cordy¹, Enrico Tronci² and Axel Legay³

¹University of Luxembourg - SnT, 6 Rue Richard Coudenhove-Kalergi, 1359 Luxembourg

²Sapienza University of Rome, Piazzale Aldo Moro, 5, 00185 Roma RM, Italia

³UCLouvain, Pl. de l'Université 1, 1348 Ottignies-Louvain-la-Neuve, Belgium

Abstract

Cyber-Physical Systems (CPS) are dynamic systems in which hardware and software components are interconnected and currently they are one of the most fundamental element of the Industry 4.0. Commonly, since the original CPS is not available for experiments, it is adopted a model of the CPS under analysis in order to execute realistic simulations on it by means of particular tools and techniques. Considering that the aim of such models and their simulation is to reproduce a real system having its authentic behavior, it is pertinent to assume that various components may be impacted by various type of uncertainties. It is relevant to take these latter into account as they may affect the system to different extents due to both the selected configuration and the simulation scenarios. The analysis of CPS signals allows the comprehension of the relationships that determinate the behavior of the entire system. When uncertainties occur, according to the chosen scenarios and context, the outcomes of simulations may be very different numerical values from the ones obtained as results of simulations without uncertainties. Nevertheless, in case of a CPS having high variability and configurability, the simulations with additional uncertainties are particularly complex to be elaborated and analyzed. Given a set of scenarios, the pursued procedure inspects the validation of possible cross-configurations, in order that the solution contains sets of appropriate configurations that takes into account both the CPS and the uncertainties.

Keywords

Simulation, Cyber-Physical Systems, Verification

1. Introduction

Cyber-Physical Systems (CPS) are denoted by the combination of physical and computational processes whose design is profoundly integrated between the cyber component and the physical one. In addition to this and conversely to desktop computing, wireless sensor network, as well as standard embedded/real-time systems, CPS have some specific characteristics including: 1) adaptive abilities (e.g. dynamic reconfiguration), 2) automation (e.g. advanced feedback control), 3) reliable operations (e.g. certified activities in terms of safety and security), 4) complex and


*Corresponding author.

✉ valeria.trombetta@uni.lu (V. Trombetta); maxime.cordy@uni.lu (M. Cordy); tronci@di.uniroma1.it (E. Tronci); axel.legay@uclouvain.be (A. Legay)

🆔 0000-0001-9795-0966 (V. Trombetta); 0000-0001-8312-1358 (M. Cordy); 0000-0002-0377-3119 (E. Tronci); 0000-0003-2287-8925 (A. Legay)



© 2022 Copyright for this paper by its authors. Use permitted under Creative Commons License Attribution 4.0 International (CC BY 4.0).

 CEUR Workshop Proceedings (CEUR-WS.org)

various spatial and temporal scales (i.e. CPS execution is bounded by spatiality and real time), 5) scalability, 6) limited resources and the presence of the software in each physical component and embedded system [1]. CPS can concern various domains including aerospace [2], automotive [3], avionic [4], healthcare [5], transportation [6], industrial production [7, 8], environmental [9], UAV [10] etc. Despite the domains to which a CPS can belong, commonly all these kind of systems have elements of variability, which, for the context of this work, refers to presence of at least one component that is not the same of the original CPS. Various CPS have to deal with multiple contexts that incorporate new customer requirements and changing environmental conditions. Each time that an unseen context occurs, the development of a new variant of a system totally from scratch becomes costly and inefficient. Normally a better adaptability, compliance and economies of scale, can be gained by engineers inserting variation points in the system, in order that the latter can then be adjusted to specific new contexts. Consequently Variability-Intensive Systems (VIS) is the name of such systems. VIS include a huge class of systems that can result into multiple variants including software product lines ([11]) and configurable systems ([12][13]). The notion of variability refers to all the way in which the variants are dissimilar, comprising, for instance, a variable having a different value. Commonly these variation points are known as features, and as parameters in software product lines and in configurable systems respectively. Considering the selected scenarios, engineers have to develop, deploy and run adequate variants of their VIS, to provide the satisfaction of the requirements. To pursue this aim, they commence the development of a configuration process in which the configuration parameters (i.e. the variation points) of the VIS are set to precise values to get the appropriate variant ([12][14]. In various embedded systems (i.e. real-world cases), the variability has an impact on the behaviour of the system in multiple fashions. As consequence of this situation, it is not banal to evaluate the variants taking into account the expected requirements. It is more evident in large systems (i.e. system with various sources of variability) how the presence of variability affects the system behavior. Typically variability is taken into account only in huge systems, however in some cases it could be essential to monitor also the one related to small systems as computationally even this one could become extremely expensive to be analyzed, depending on the specific contexts and scenarios. This raised the first three research questions: **RQ1) Given a scenario, is it fundamental to monitor the effect of variability even in small systems? RQ2) What is the total size of configuration space composed by multiple small configuration systems and their scenarios? RQ3) Can we reduce this complex configuration space to a smaller step of solutions?** Additionally to what aforementioned, the injection of variability may get in an intractable situation because of the exponential computation due to many millions of variants ([15]). Moreover considering that VIS may belong to several domains, the identification of adequate variants is even more complex due to the presence of multiple constraints and criteria to be taken into account. Furthermore the selection of the appropriate variant becomes even more challenging when uncertainties (e.g. noises, disturbances, etc.) occur. This is due to the fact that the requirements of the system should be satisfied with the highest probability in the majority of contexts ([16][17]) but in these situations the system behaves unpredictably. From this, our second research question is: **RQ4) Do the identified best configurations remain valid in the presence of uncertainties?** As consequence of the presence of uncertainties and configurability, to validate each configuration it is useful to monitor if they had an impact on the success of the mission taking into account

the requirements. From here, the fifth research question is: **RQ5) Given a set of scenarios, can we identify the optimal configuration wrt the budget and the desired assurance level?** In VIS the variability can be related to the design-time and the run-time. At design-time engineers have already stated the definition of the requirements, so the purpose is to detect which variants of their VIS are more likely to satisfy the requirements in the best possible way and then which ones should be developed/deployed. At run-time, unpredictable changes in the environment are present to have an impact on the already running system (i.e. a specific variant of the VIS). To verify that the requirements are still satisfied in spite of the mutable ambient situations, engineers have to reconfigure the system, i.e. swapping the latter from a variant to another one by changing the value of its configuration parameters during its execution. This is why the development of an approach that can manage this two kind of variability is complex, and this brings the sixth research question: **RQ6) Do the identified configurations remain valid taking into account both the design-time and run-time variability?** The preliminary work introduced in this paper is targeted on the analysis of Cyber-Physical Systems (CPS) [18, 19] with high variability and intensity, and their related behavior in presence of variability concerning parameter configurations and uncertainties. The aim of the research is to help engineers to investigate all the possible and appropriate configuration alternatives at both design and run-time wrt the scenario, the mission, the assurance level, and the budget. This raised the seventh research question: **RQ7) Given a set of scenarios, an assurance level, a budget whose expenditure must be minimised, is the identified configuration the same for both design and run-time?** Taking into account that this type of analysis is extremely time consuming as based on highly realistic simulations and models, this is the eighth research question that has been identified: **RQ8) Is it feasible to develop an approach to reduce the computational time related to the simulations?** To answer to all these research questions we modeled a set of scenarios, uncertainties, and a system based on UAV, more precisely on a swarm of drones, which are extremely critical and are intensively susceptible to environmental perturbations. In general as a consequence of their critical missions it is essential that CPS behave appropriately and do not act unexpectedly even in case of uncertainties. This is why the design, construction and verification of such systems are significant phases throughout which quality should be confirmed. However these phases are challenging due to the difficulty to represent and reproduce the various and complex system-world interactions [20, 21, 22]. The quality is also a fundamental component to be demonstrated as it concerns the satisfaction of various safety industrial standards (e.g. ISO). Taking into account a mission and a scenario the satisfaction of safety standards (i.e. the quality) is the core component to develop a system that could exist in reality. CPS engineers typically rely on simulations and related platforms, such as MathWorks' software products (e.g. MATLAB, Simulink, etc.), to prepare and assess candidate designs for their system. MATLAB and Simulink are de-facto industry standard [23][24] well known by engineers. MATLAB is a programming language as well as the tool that contains Simulink, an environment based on block diagrams which permits the modeling and simulations of dynamical systems. The case study we modeled can be customized to run extra experiments in an uncomplicated way. The framework developed to perform preliminary analysis and experiments is easy-reusable and extendable. Such framework and the model are also totally developed adopting the notorious MathWorks software products, so it is not necessary to engineers to gain expertise with extra frameworks.

2. Research Methodology and Approach

The principal aim consists in the identification of the best configuration in terms of accomplishment of the mission, the scenario, the possible presence of uncertainties, the assurance level, as well as the costs that have to be minimized. In addition to this we investigated if the injection of various kind of uncertainties lead to a change of the configuration wrt the same scenario in which uncertainties are absent. Furthermore we analyzed if the choice of the type (i.e. the quality) of drones have a major impact on the success of the mission or if the latter depends on the amount of drones in swarms, or if such kind of selection depends on the specific scenarios. To reach such an objective, we modeled a case study based on a swarm of drones in which each drone has two features related to the nature of the battery and the radio. Both of these features can have three different values (i.e. top, medium, low) which represent the quality of that component. For example, a top battery has a higher capacity than the other ones, and a top radio affects to a lesser extent the batteries. Internally the radio has two modules namely the transmission and the reception ones. This latter is modeled to be always on and has a limited impact on the battery. Conversely to this the transmission module consumes more battery and it is designed to be active only for the first 5 seconds of the simulations. The reason of this is to represent the fact that at the beginning of simulations ideally drones are receiving the coordinates related to the initial position of the target, and drones are supposed to send back an "ack" signal. If at least a drone of the swarm reaches the target, the mission is accomplished. To prevent collisions, the target is considered reached if a drone reach it having a meter of distance. The target is always present and unique. In addition to this it can be fixed, or in movement adopting a random motion, depending on the considered scenario. If a drone is almost out of battery, it lands. Drones do not communicate each other, and in the swarm there is not any "leader" drone. The modeled arena has fixed dimensions, and on it the first drone of the swarm spawns in a random location. According to this position all the other drones spawn randomly in a range of 3 meters. The reason of this choice is to prevent drones to be located in positions that are too far from each other, as in this way the perception of the swarm would be lost. All the drones spawn at the same time and they all have anti-collisions mechanisms [25]. The target spawns in a random location having the constraint to appear in the opposite part of the arena wrt the swarm position. This is due to prevent to have simulations in which the swarm and the target are too close or even overlapped, as this would have get an unnecessarily over-complicated analysis of the behavior of the swarm, drones battery, etc., to avoid to have unrealistic scenarios, as well as to force the swarm to pass through a group of obstacles. In some scenarios in fact it has been modeled a group of 50 obstacles that spawns in random locations of the arena. If present, obstacles can be fixed or moving having a random motion. Both the target and the obstacles do not change their dynamics: if they are fixed, they remain stationary for the entire simulation, and vice-versa in case they are moving. The drones have the same anti-collision approach also wrt to the obstacles but such condition is not mutual. In case of scenario with enabled faults, both the modules of drones radio may be disabled/activated in multiple moments and time ranges during simulations. Faults are modeled as a Markov Chain [26][27] having 2 states and equal probability on all the edges. The two states of the Markov Chain represent the correct/incorrect behavior of the system, and at every second of the simulation a state transition may occur. Faults affect the behavior of the radio

(e.g. the transmission module of the radio may be enabled for various unnecessary moments and this affects the level of the battery). As additional forms of uncertainties white noises [28] and random gusts are modeled on drones thrusters, and on drones trajectories respectively. If enabled, uncertainties are present for all the drones of the swarms. In order to have a symbolic representation of the costs for the swarm, we applied 8, 5, 2 for the batteries and 3, 2, 1 for the radio, where the higher numbers represent the top quality. We decided to assign higher values to the battery as empirically we noticed that it is the most relevant element that characterizes the quality of the drone wrt the success of the mission.

3. Preliminary Results

The swarm of drones is entirely modeled adopting MATLAB as programming language, and the simulations are performed in MATLAB environment. Such simulations are the Monte Carlo ones [29][30][31]. In our evaluations we considered as negligible the random locations in which drones spawn, the one of the target, as well as the ones of obstacles if these latter are present. We also considered irrelevant the possibility that multiple drones arrive close to the target at the same time and that there will be a consequent effect on motion due to the anti-trajectory approach. The assurance level that we decided to consider is 96%, which means that the discovered configurations ensure that at least a drone of the swarm reaches the target with that probability. To answer to **RQ1**) we empirically proved that even for small systems having just two features, it is fundamental to monitor the variability as the latter may have a decisive impact on the accomplishment of the mission (i.e. reaching the target), especially when in the swarm there are few drones. In addition to this, in general the total effect of such small systems is embedded in complex scenarios which further increase the variability management. Pondering that drones can have 9 different configurations, the mix on the quality of battery and radio, the considered maximum amount of drones in the swarm that we took into account for our preliminary experiments which is 9, all the possible scenarios in terms of type of target, obstacles, and uncertainties, the total size of configuration space is 12264 and this is the answer for **RQ2**). About the **RQ3**), it is possible to reduce the complex configuration space to a smaller one only when in the swarm there are drones of the same type (e.g. only top quality drones), but even if such kind of configurations would accomplish the mission having the aforementioned assurance level, one of the main goals of this work consists in the identification of suitable configurations that minimize the costs and satisfy the requirements in terms of the success of the mission and the assurance level. The selection of a swarm composed exclusively by top drones would not permit the costs to be minimized as trivially low-medium quality drones are less pricey. For all the scenarios we discovered in fact configurations that allow to save the budget maintaining the same assurance level on the success of the mission. On **RQ4**) for each scenario we determined that only a type of uncertainties had an impact on the choice of the configuration of the swarms. For instance we discovered that in no scenario moving obstacles, noises and faults lead to a change of the configuration, namely considering two versions (i.e. with and without noises and faults) of the same scenario, the identified configuration remains the same, conversely to scenarios in which gusts are present. If these latter are enabled we discovered that for any scenario it is essential to have better quality drones

in order to accomplish the mission. The enhanced quality of drones is not a valid solution in every case, in fact performing experiments we determined that the main element that affect the configuration selection configuration is related to the presence of the mobile target. In such kind of scenario the best configurations have an additional amount of drones. So the two identified solutions to mitigate with uncertainties (i.e. gusts and the aleatoriness related to the random motion of the target) are not equivalent. For each scenario when the gusts are present on average the costs increases by 5, and it increases by 7 in case of moving target. To answer **RQ5**) for all scenarios we have identified the configuration that satisfies the assurance level, minimize the costs wrt the budget, permits the success of the mission. None of these identified configurations include a swarm with all top drones or with the maximum amount of drones that for the considered preliminary experiments is 9. **RQ6**), **RQ7**), **RQ8**) are part of future development. Some threats to the validity of the current work include the fact that in order to evaluate our results we performed 50 Monte Carlo simulations for every cross-configuration on swarms and the scenarios. Despite all the identified configurations have 96% as assurance level in terms of success of the mission, we cannot state if the same configurations would have been selected performing further Monte Carlo simulations. In addition to this, even if we run very realistic simulations, we did considered yet simulation with specific ISO standards, so we cannot state if these design choices could be applied in reality (e.g. due to constraints related to standards, etc.) for the scenarios taken into account. The purpose of this preliminary work was to show the existence of a huge variability space even in small systems like drones, and that such variability can become even more challenging to be evaluated due to the complexity that grows up exponentially when combined with additional small systems (i.e. the swarms) as well as in realistic scenarios. This is why also variability related to small systems have to be deeply analyzed and investigated. Moreover as contribution we modeled a case study including the model, uncertainties and a set of scenarios, which can be customized and extended, as well as a framework that can be easily tailored for additional models and re-used for further type of experiments and analysis. It will be part of future work to validate the experiments considering various safety standards, different simulations settings, as well as additional missions, scenarios and run-time variability.

4. Preliminary Results

The swarm of drones is entirely modeled adopting MATLAB as programming language, and the simulations are performed in MATLAB environment. Such simulations are the Monte Carlo ones [29][30][31]. In our evaluations we considered as negligible the random locations in which drones spawn, the one of the target, as well as the ones of obstacles if these latter are present. We also considered irrelevant the possibility that multiple drones arrive close to the target at the same time and that there will be a consequent effect on motion due to the anti-trajectory approach. The assurance level that we decided to consider is 96%, which means that the discovered configurations ensure that at least a drone of the swarm reaches the target with that probability. To answer to **RQ1**) we empirically proved that even for small systems having just two features, it is fundamental to monitor the variability as the latter may have a decisive impact on the accomplishment of the mission (i.e. reaching the target), especially

when in the swarm there are few drones. In addition to this, in general the total effect of such small systems is embedded in complex scenarios which further increase the variability management. Pondering that drones can have 9 different configurations, the mix on the quality of battery and radio, the considered maximum amount of drones in the swarm that we took into account for our preliminary experiments which is 9, all the possible scenarios in terms of type of target, obstacles, and uncertainties, the total size of configuration space is 12264 and this is the answer for **RQ2**). About the **RQ3**), it is possible to reduce the complex configuration space to a smaller one only when in the swarm there are drones of the same type (e.g. only top quality drones), but even if such kind of configurations would accomplish the mission having the aforementioned assurance level, one of the main goal of this work consists in the identification of suitable configurations that minimize the costs and satisfy the requirements in terms of the success of the mission and the assurance level. The selection of a swarm composed exclusively by top drones would not permit the costs to be minimized as trivially low-medium quality drones are less pricey. For all the scenarios we discovered in fact configurations that allow to save the budget maintaining the same assurance level on the success of the mission. On **RQ4**) for each scenario we determined that only a type of uncertainties had an impact on the choice of the configuration of the swarms. For instance we discovered that in no scenario moving obstacles, noises and faults lead to a change of the configuration, namely considering two versions (i.e. with and without noises and faults) of the same scenario, the identified configuration remains the same, conversely to scenarios in which gusts are present. If these latter are enabled we discovered that for any scenario it is essential to have better quality drones in order to accomplish the mission. The enhanced quality of drones is not a valid solution in every case, in fact performing experiments we determined that the main element that affect the configuration selection configuration is related to the presence of the mobile target. In such kind of scenario the best configurations have an additional amount of drones. So the two identified solutions to mitigate with uncertainties (i.e. gusts and the aleatoriness related to the random motion of the target) are not equivalent. For each scenario when the gusts are present on average the costs increases by 5, and it increases by 7 in case of moving target. To answer **RQ5**) for all scenarios we have identified the configuration that satisfies the assurance level, minimize the costs wrt the budget, permits the success of the mission. None of these identified configurations include a swarm with all top drones or with the maximum amount of drones that for the considered preliminary experiments is 9. **RQ6**), **RQ7**), **RQ8**) are part of future development. Some threats to the validity of the current work include the fact that in order to evaluate our results we performed 50 Monte Carlo simulations for every cross-configuration on swarms and the scenarios. Despite all the identified configurations have 96% as assurance level in terms of success of the mission, we cannot state if the same configurations would have been selected performing further Monte Carlo simulations. In addition to this, even if we run very realistic simulations, we did considered yet simulation with specific ISO standards, so we cannot state if these design choices could be applied in reality (e.g. due to constraints related to standards, etc.) for the scenarios taken into account. The purpose of this preliminary work was to show the existence of a huge variability space even in small systems like drones, and that such variability can become even more challenging to be evaluated due to the complexity that grows up exponentially when combined with additional small systems (i.e. the swarms) as well as in realistic scenarios. This is why also variability related to small systems have to be deeply

analyzed and investigated. Moreover as contribution we modeled a case study including the model, uncertainties and a set of scenarios, which can be customized and extended, as well as a framework that can be easily tailored for additional models and re-used for further type of experiments and analysis. It will be part of future work to validate the experiments considering various safety standards, different simulations settings, as well as additional missions, scenarios and run-time variability.

5. Work Plan

As future directions for the project, the idea is to answer to all the research questions performing experiments on additional systems, including the ones belonging to other domains, such as swarms of Cubesats having different roles and characteristics for specific missions as well as group of aircrafts, etc. Considering the modeled case study we plan to validate experiments wrt to various specific safety industrial standards, additional missions and scenarios, and different uncertainties. We also plan to run experiments to identify the best configurations also wrt KPIs such as the final level of battery when the target is reached in order to evaluate the configurations also according to further criteria that can be considered relevant for specific missions (e.g. if the goal is to get to the target and then to come back at the starting point, the level of battery when the target is reached, is a relevant KPI). About the target the presence of multiple targets will be modeled as well as the addition of a "leader" drone that will be in charge to reach one or more targets. Moreover additional missions design will include the presence of swarms having more drones, as well as the adoption of other motions for the target and the obstacles. In furtherance of this, for example, if the leader drone has some faults, the swarm can be reconfigured to have a new leader, as well as if any "non-leader" drone is closer to a target than the current leader. This is what we plan to develop to answer to **RQ6** as well as to compare our approach with the already existing ones. On **RQ7**, in addition to the modeling of missions and scenarios in which run-time configuration will be included, we plan to adopt simulations based on the Empirical Bernstein Stopping Algorithm [32]. In this way for each scenarios simulations will be automatically stopped by the algorithm when the suitable configuration will be identified rather than after an a-priori defined about of runs. This in conjunction with the consideration of ISO standards will solve the aforementioned threats to the validity. Concerning **RQ8**, currently it is under development an approach based on simulation snapshots. The plan consists in establishing similarity thresholds that monitors the values of variables belonging to different configurations, if these values are equal or less than the established threshold, the simulation is stopped. Since simulations are extremely time consuming, the idea is to avoid to run entirely all the simulations related to configurations that are too similar and at the same time to explore all the configuration space. For the following months this will be the adopted working plan: *Sep.-Oct.* : enhancement and automatization of the snapshot approach *Nov.-Dec.* : experiments and comparison with other approaches. *Jan.-Feb.* : experiments on additional models (e.g. industrial models belonging to other domains or multi-domains, etc.) and adoption of safety standards *Mar-Apr.* : addition of further scenarios, missions, adoption of different simulation setting. *May-Jun.* : to run experiments on run-time variability and comparison with other approaches for such type of systems.

6. Related Work

All of the following works make use of MathWorks Software Products for variability modeling, but they mainly address the system configurability and do not consider any uncertainties. Alalfi et al. [33] have empirically derived five variability operators for Simulink models. Leitner et al. [34] have enhanced the variability by adopting layers of abstractions and an extra binding time for Simulink models. According to the survey elaborated by Berger et al. [35] as many as 38% of respondents have used a home-grown domain-specific tool, including Simulink, to perform activities related to the variability modeling in industrial practice. Schlie et al. [36] proposed an holistic approach for the reengineering of an entire Simulink model portfolio into a single variability model. Schulze et al. [37] described the problem of intermixing of various function variants with the variability switching logic adopting a Simulink model. Arrieta et al. [38] proposed a methodology for mutating configurable Simulink models where their variability is expressed as feature models. Finally, Haber et al. [39] applied the method of delta modeling [40] to the Simulink environment in order to obtain a modular and flexible variability modeling approach suitable for Simulink models. Regarding the use of Simulink and thus also MATLAB as tools, authors such as Bressan et al. [41], who developed a tool for specifying variability in safety-critical systems and which can produce the correct system configuration models. On the modeling of swarms of drones adopting MATLAB, Soria et al. [42] modeled a swarm analyzing some performances, however they do not consider uncertainties and different kind of drones. Lee et al. [43] designed endogenous paradigms to monitor effects on the increased degree of freedom on a swarm. Quesada et al [44] adopted fuzzy logic theory for the generation of leader-follower behavior in a swarm. Other related works are those regarding the application of software product line (SPL) methods [45] to CPS. The SPL engineering paradigm promotes systematic reuse and a-priori identification of variation points in order to make the development of software variants more effective. Our work does not try to adapt SPL to CPS models in the UAV field but uses MATLAB, a notorious tool already used in both SPL and the UAV domain, to model and analyze the effect of variability using a cross-configurations approach. To the best of our knowledge, there are not other methods oriented to the analysis of the configuration of both CPS elements and uncertainties, because all the surveyed approaches focus only on the first one. On run-time and Software Product Lines Cetina et al. [46] performed an evaluation about reconfigurations and related reliability-based risks, more precisely, the availability and the severity. Van der Hoek [47] developed an infrastructure with which variabilities can be specified at design time, but resolved at any time thereafter. Gomaa and Hussein [48] delineated a methodology based on architecture patterns to design software reconfiguration patterns. The State-of-the art on Behavioural Verification of VIS include Model checking [49] that represent one of the most notorious technique to analyse system behaviour wrt requirements. Typically, it takes as inputs a state machine (i.e. an executable model of the system) to be verified and a logic formula that contains the requirements that the system is expected to satisfy, and it outputs false and counterexamples if the executions of the model did not satisfy the logic formula, true otherwise. The model-checking problem for VIS is, more complex than for single systems since each variant must be verified wrt the formula [50]. A possible solution consists in adopting a single-system model checking to all the variants in a separate way. These procedures are called product-based in the software-product-line jargon [51], but due to their monitor of each

variant individually they may lead to the exponential blow-up induced by variability. To avoid this complexity, family-based [51] model-checking approaches were developed [50][52]. Their objective is based on the decrement of the verification effort by taking into consideration the commonalities that are present in multiple variants. The analysis is executed on each variant at the same time, using approaches such as late splitting and early joining to check only once a (part of) execution common to multiple variants. In this way they decrement the exponential blow-up, despite it can still be present [53][54]. Feature-based model checking[55] [56] is based on the same aim of avoiding computations that are not indispensable. These approaches consider that variation points are compositional and decrease the analysis of one variant to the individual analysis of its variation points. This consideration is valid only for VIS that are structured in particular fashions and do not generalize. Family-based and feature-based approaches were taken into account to analyse the behaviour of stochastic VIS [16][17], but they are not scalable or they have strict assumptions on how variation points can affect the system, and this is valid only in few particular cases. Sample-based methods are the trade-off between the product-based and feature-based ones [51] [57]. They analyse variants separately but they consider also a degraded form of compositionality across features such that the results for the sampled variants can facilitate the inference of the characteristics of the non-sampled ones. Such goal is reached since the behaviour of variants that are not known is already considered by the ones that are known or via extrapolation based on prediction models [58][59][60].

Acknowledgments

This project is supported by FNR Luxembourg (grant C19/IS/13566661/BEEHIVE/Cordy).

References

- [1] J. Shi, J. Wan, H. Yan, H. Suo, A survey of cyber-physical systems, in: 2011 international conference on wireless communications and signal processing (WCSP), IEEE, 2011, pp. 1–6.
- [2] L. Zhang, View oriented approach to specify and model aerospace cyber-physical systems, in: 2013 IEEE 11th International Conference on Dependable, Autonomic and Secure Computing, IEEE, 2013, pp. 296–303.
- [3] A. Patil, K. More, S. Kulkarni, A review on vehicular cyber physical systems, International Research Journal of Engineering and Technology (IRJET) 5 (2018) 1138–1143.
- [4] K. Sampigethaya, R. Poovendran, Aviation cyber-physical systems: Foundations for future aircraft and air transport, Proceedings of the IEEE 101 (2013) 1834–1855.
- [5] S. A. Haque, S. M. Aziz, M. Rahman, Review of cyber-physical system in healthcare, international journal of distributed sensor networks 10 (2014) 217415.
- [6] L. Deka, S. M. Khan, M. Chowdhury, N. Ayres, Transportation cyber-physical system and its importance for future mobility, in: Transportation Cyber-Physical Systems, Elsevier, 2018, pp. 1–20.
- [7] P. J. Mosterman, J. Zander, Industry 4.0 as a cyber-physical system study, Software & Systems Modeling 15 (2016) 17–29.

- [8] L. Wang, M. Törngren, M. Onori, Current status and advancement of cyber-physical systems in manufacturing, *Journal of Manufacturing Systems* 37 (2015) 517–527.
- [9] G. Mois, T. Sanislav, S. C. Folea, A cyber-physical system for environmental monitoring, *IEEE Transactions on Instrumentation and Measurement* 65 (2016) 1463–1471.
- [10] R. Shakeri, M. A. Al-Garadi, A. Badawy, A. Mohamed, T. Khattab, A. K. Al-Ali, K. A. Harras, M. Guizani, Design challenges of multi-uav systems in cyber-physical applications: A comprehensive survey and future directions, *IEEE Communications Surveys & Tutorials* 21 (2019) 3340–3385.
- [11] L. Northrop, Sei's software product line tenets, *IEEE Software* 19 (2002) 32–40. doi:10.1109/MS.2002.1020285.
- [12] D. Sabin, R. Weigel, Product configuration frameworks-a survey, *IEEE Intelligent Systems and their applications* 13 (1998) 42–49.
- [13] M. Raatikainen, T. Soininen, T. Männistö, A. Mattila, A case study of two configurable software product families, in: *International Workshop on Software Product-Family Engineering*, Springer, 2003, pp. 403–421.
- [14] M. Cordy, P. Heymans, Engineering configurators for the retail industry: experience report and challenges ahead, in: *Proceedings of the 33rd Annual ACM Symposium on Applied Computing*, 2018, pp. 2050–2057.
- [15] T. Berger, S. She, R. Lotufo, A. Wąsowski, K. Czarnecki, Variability modeling in the real: a perspective from the operating systems domain, in: *Proceedings of the IEEE/ACM international conference on Automated software engineering*, 2010, pp. 73–82.
- [16] G. N. Rodrigues, V. Alves, V. Nunes, A. Lanna, M. Cordy, P.-Y. Schobbens, A. M. Sharifloo, A. Legay, Modeling and verification for probabilistic properties in software product lines, in: *2015 IEEE 16th International Symposium on High Assurance Systems Engineering*, IEEE, 2015, pp. 173–180.
- [17] L. Pessoa, P. Fernandes, T. Castro, V. Alves, G. N. Rodrigues, H. Carvalho, Building reliable and maintainable dynamic software product lines: An investigation in the body sensor network domain, *Information and Software Technology* 86 (2017) 54–70.
- [18] A. Platzer, *Logical Foundations of Cyber-Physical Systems*, 1st ed., Springer Publishing Company, Incorporated, 2018.
- [19] L. Hu, N. Xie, Z. Kuang, K. Zhao, Review of cyber-physical system architecture, in: *2012 IEEE 15th international symposium on object/component/service-oriented real-time distributed computing workshops*, IEEE, 2012, pp. 25–30.
- [20] A. Gerstlauer, C. Haubelt, A. D. Pimentel, T. P. Stefanov, D. D. Gajski, J. Teich, Electronic system-level synthesis methodologies, *IEEE Trans. Comput. Aided Des. Integr. Circuits Syst.* 28 (2009) 1517–1530.
- [21] S. A. Seshia, S. Hu, W. Li, Q. Zhu, Design automation of cyber-physical systems: Challenges, advances, and opportunities, *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems* 36 (2016) 1421–1434.
- [22] S. Bliudze, S. Furic, J. Sifakis, A. Viel, Rigorous design of cyber-physical systems, *Software & Systems Modeling* 18 (2019) 1613–1636.
- [23] B. Murgante, O. Gervasi, S. Misra, N. Nedjah, A. M. A. C. Rocha, D. Taniar, B. O. Apduhan, *Computational Science and Its Applications–ICCSA 2012: 12th International Conference*, Salvador de Bahia, Brazil, June 18–21, 2012, Proceedings, Part I, volume 7333, Springer,

2012.

- [24] S. L. Shrestha, S. A. Chowdhury, C. Csallner, Deepfuzzsl: Generating models with deep learning to find bugs in the simulink toolchain, in: 2nd Workshop on Testing for Deep Learning and Deep Learning for Testing (DeepTest), 2020.
- [25] C. R. Stark, L. M. Pyke, Dynamic pathfinding for a swarm intelligence based uav control model using particle swarm optimisation, *Frontiers in Applied Mathematics and Statistics* 7 (2021) 744955.
- [26] S. Brooks, Markov chain monte carlo method and its application, *Journal of the royal statistical society: series D (the Statistician)* 47 (1998) 69–100.
- [27] C. J. Geyer, Practical markov chain monte carlo, *Statistical science* (1992) 473–483.
- [28] H.-H. Kuo, *White noise distribution theory*, CRC press, 2018.
- [29] C. Z. Mooney, *Monte carlo simulation*, 116, Sage, 1997.
- [30] E. Zio, Monte carlo simulation: The method, in: *The Monte Carlo simulation method for system reliability and risk analysis*, Springer, 2013, pp. 19–58.
- [31] R. L. Harrison, Introduction to monte carlo simulation, in: *AIP conference proceedings*, volume 1204, American Institute of Physics, 2010, pp. 17–21.
- [32] V. Mnih, C. Szepesvári, J.-Y. Audibert, Empirical bernstein stopping, in: *Proceedings of the 25th international conference on Machine learning*, 2008, pp. 672–679.
- [33] M. H. Alalfi, E. J. Rapos, A. Stevenson, M. Stephan, T. R. Dean, J. R. Cordy, Variability identification and representation for automotive simulink models, in: *Automotive Systems and Software Engineering*, Springer, 2019, pp. 109–139.
- [34] A. Leitner, W. Ebner, C. Kreiner, Mechanisms to handle structural variability in matlab/simulink models, in: *International Conference on Software Reuse*, Springer, 2013, pp. 17–31.
- [35] T. Berger, R. Rublack, D. Nair, J. M. Atlee, M. Becker, K. Czarnecki, A. Wasowski, A survey of variability modeling in industrial practice, in: *Proceedings of the Seventh International Workshop on Variability Modelling of Software-intensive Systems*, 2013, pp. 1–8.
- [36] A. Schlie, C. Seidl, I. Schaefer, Reengineering variants of matlab/simulink software systems, in: *Security and Quality in Cyber-Physical Systems Engineering*, Springer, 2019, pp. 267–301.
- [37] M. Schulze, J. Weiland, D. Beuche, Automotive model-driven development and the challenge of variability, in: *Proceedings of the 16th International Software Product Line Conference-Volume 1*, 2012, pp. 207–214.
- [38] A. Arrieta, U. Markiegi, L. Etxeberria, Towards mutation testing of configurable simulink models: a product line engineering perspective, *Jornadas de Ingeniera del So ware y Bases de Datos (JISBD)* (2017).
- [39] A. Haber, C. Kolassa, P. Manhart, P. M. S. Nazari, B. Rumpe, I. Schaefer, First-class variability modeling in matlab/simulink, in: *Proceedings of the Seventh International Workshop on Variability Modelling of Software-intensive Systems*, 2013, pp. 1–8.
- [40] D. Clarke, M. Helvensteijn, I. Schaefer, Abstract delta modelling, *Mathematical Structures in Computer Science* 25 (2015) 482–527.
- [41] L. Bressan, A. L. de Oliveira, F. Campos, R. Capilla, A variability modeling and transformation approach for safety-critical systems, in: *15th International Working Conference on Variability Modelling of Software-Intensive Systems*, 2021, pp. 1–7.

- [42] E. Soria, F. Schiano, D. Floreano, Swarmlab: A matlab drone swarm simulator, in: 2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), IEEE, 2020, pp. 8005–8011.
- [43] D.-H. Lee, J.-H. Jeong, H.-J. Ahn, S.-W. Lee, Design of an eeg-based drone swarm control system using endogenous bci paradigms, in: 2021 9th International Winter Conference on Brain-Computer Interface (BCI), IEEE, 2021, pp. 1–5.
- [44] W. O. Quesada, J. I. Rodriguez, J. C. Murillo, G. A. Cardona, D. Yanguas-Rojas, L. G. Jaimes, J. M. Calderón, Leader-follower formation for uav robot swarm based on fuzzy logic theory, in: International Conference on Artificial Intelligence and Soft Computing, Springer, 2018, pp. 740–751.
- [45] R. Rabiser, A. Zoitl, Towards mastering variability in software-intensive cyber-physical production systems, *Procedia Computer Science* 180 (2021) 50–59.
- [46] C. Cetina, P. Giner, J. Fons, V. Pelechano, Prototyping dynamic software product lines to evaluate run-time reconfigurations, *Science of Computer Programming* 78 (2013) 2399–2413.
- [47] A. Van der Hoek, Design-time product line architectures for any-time variability, *Science of computer programming* 53 (2004) 285–304.
- [48] H. Gomaa, M. Hussein, Software reconfiguration patterns for dynamic evolution of software architectures, in: Proceedings. Fourth Working IEEE/IFIP Conference on Software Architecture (WICSA 2004), IEEE, 2004, pp. 79–88.
- [49] F. Laroussinie, Christel baier and joost-pieter katoen principles of model checking. mit press (may 2008). isbn: 978-0-262-02649-9. 44.95. 975 pp. hardcover, *The Computer Journal* 53 (2010) 615–616.
- [50] A. Classen, P. Heymans, P.-Y. Schobbens, A. Legay, J.-F. Raskin, Model checking lots of systems: efficient verification of temporal properties in software product lines, in: Proceedings of the 32nd ACM/IEEE International Conference on Software Engineering-Volume 1, 2010, pp. 335–344.
- [51] T. Thüm, S. Apel, C. Kästner, I. Schaefer, G. Saake, A classification and survey of analysis strategies for software product lines, *ACM Computing Surveys (CSUR)* 47 (2014) 1–45.
- [52] P. Asirelli, M. H. Ter Beek, S. Gnesi, A. Fantechi, Formal description of variability in product families, in: 2011 15th International Software Product Line Conference, IEEE, 2011, pp. 130–139.
- [53] A. Classen, M. Cordy, P.-Y. Schobbens, P. Heymans, A. Legay, J.-F. Raskin, Featured transition systems: Foundations for verifying variability-intensive systems and their application to ltl model checking, *IEEE Transactions on Software Engineering* 39 (2012) 1069–1089.
- [54] M. Cordy, A. Classen, P. Heymans, A. Legay, P.-Y. Schobbens, Model checking adaptive software with featured transition systems, in: Assurances for Self-Adaptive Systems, Springer, 2013, pp. 1–29.
- [55] M. Plath, M. Ryan, Feature integration using a feature construct, *Science of Computer Programming* 41 (2001) 53–84.
- [56] K. Fisler, S. Krishnamurthi, Decomposing verification around end-user features, in: Working Conference on Verified Software: Theories, Tools, and Experiments, Springer, 2005, pp. 74–81.

- [57] S. Apel, A. Von Rhein, P. Wendler, A. Größlinger, D. Beyer, Strategies for product-line verification: case studies and experiments, in: 2013 35th International Conference on Software Engineering (ICSE), IEEE, 2013, pp. 482–491.
- [58] J. Oh, D. Batory, M. Myers, N. Siegmund, Finding near-optimal configurations in product lines by random sampling, in: Proceedings of the 2017 11th Joint Meeting on Foundations of Software Engineering, 2017, pp. 61–71.
- [59] J. Guo, D. Yang, N. Siegmund, S. Apel, A. Sarkar, P. Valov, K. Czarnecki, A. Wasowski, H. Yu, Data-efficient performance learning for configurable systems, *Empirical Software Engineering* 23 (2018) 1826–1867.
- [60] N. Siegmund, A. Grebhahn, S. Apel, C. Kästner, Performance-influence models for highly configurable systems, in: Proceedings of the 2015 10th Joint Meeting on Foundations of Software Engineering, 2015, pp. 284–294.