To What Extent Formal Methods are Applicable for Performance Analysis of Smart Cyber-Physical Systems?

Lorenzo Pagliari
Gran Sasso Science Institute, Italy
lorenzo.pagliari@gssi.it

Mirko D'Angelo
Linnaeus University, Sweden
mirko.dangelo@lnu.se

Mauro Caporuscio
Linnaeus University, Sweden
mauro.caporuscio@lnu.se

Raffaela Mirandola
Politecnico di Milano, Italy
raffaela.mirandola@polimi.it

Catia Trubiani
Gran Sasso Science Institute, Italy
catia.trubiani@gssi.it

ABSTRACT

The dynamic nature of complex Cyber-Physical Systems (CPS) introduces new research challenges since they need to smartly deal with changing situations in their environment. This triggers the usage of methodologies that keep track of changes and raise alarms whether extra-functional requirements (e.g., safety, reliability, performance) are violated. In this context, we investigate the usage of formal methods as support to provide a model-based performance evaluation of smart CPS. The main goal is to understand to what extent well-known performance models, specifically Queueing Networks, are suitable to represent these dynamic scenarios.

CCS CONCEPTS

• **Software and its engineering → Software performance**;

KEYWORDS

Cyber-Physical Systems; Queueing Networks;
Model-based Performance Analysis.

ACM Reference Format:
Lorenzo Pagliari, Mirko D'Angelo, Mauro Caporuscio, Raffaela Mirandola, and Catia Trubiani. 2019. To What Extent Formal Methods are Applicable for Performance Analysis of Smart Cyber-Physical Systems?. In *Proceedings of ACM Conference (Conference'17).* ACM, New York, NY, USA, 6 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

1 INTRODUCTION

Cyber-Physical Systems (CPS) are recently gaining momentum and attracting massive attention from research and major investments from Industry [35]. CPS are emerging in several application areas from health-care and medicine, to industry automation and manufacturing systems [19], to automotive with smart cars and intelligent road system with unmanned vehicle [28]. Such broad set of application areas makes the task of designing CPS indeed not trivial, in fact it is necessary to combine multiple components with different levels of abstraction, and it is fundamental to evaluate their extra-functional properties (e.g., security, safety, reliability, performance) [9].

In literature several approaches recently emerged for the modelling of CPS (e.g., [17, 32]). Most of them focuses on functional analysis ([32]) or verification ([17]), but performance-related characteristics result to be almost neglected. As opposite, we think that it is of key relevance to anticipate performance flaws and avoid late fixes during the testing and after the implementation in such complex systems. In fact, it has been demonstrated that the costs of fixing errors escalate in an exponential fashion as the project matures during the various phases of its life cycle [31]. Interestingly, if the cost of fixing an error during the requirements phase is estimated to be 1 unit, the cost to fix that error during the design phase increases to 3-8 units; at the manufacturing/build phase, it becomes 7-16 units; at the integration and test phases, it results to be 21-78 units, and at the operational phase, the cost ranges from 29 to more than 1500 units. This highly motivates the introduction of methodologies providing an early quantitative evaluation of software systems.

In our previous work [23] we proposed a methodology supporting the performance engineering of CPS, and we built our modeling and analysis framework on top of Ptolemy[1]. However, simulation-based analysis suffers from several issues, such as the warm-up period, stability conditions, etc. In fact, any analysis method must be statistically significant, usually a large number of replications of simulation experiments is required to determine the fraction of those experiments producing the final confidence intervals, thus to get the true mean value of the estimated parameters [7].

Goal of this paper is to focus on the application of formal methods for an early performance evaluation of CPS, specifically at design time. In particular, we intend to explore the effectiveness of these methods in terms of model representativeness and reliability of the obtained prediction results. This is helpful for software architects that need to evaluate design alternatives and performance requirements, such as the system response time has to be not larger than 5 seconds, or the resource utilization lower than 80%.

To this end, we investigate, through a simple, yet powerful case study, how formal models widely applied in the past for the performance prediction and evaluation of complex systems, like Queueing Networks (QN) [18] can fit smart CPS. To validate the model and the results, we compare QN-based prediction results with simulation values that are obtained with an ad-hoc framework. The

Conference'17, July 2017, Washington, DC, USA

ACM ISBN 978-x-xxxx-xxxx-x/YY/MM.
https://doi.org/10.1145/nnnnnnn.nnnnnnn

[1] http://ptolemy.eecs.berkeley.edu/ptolemyII/index.htm

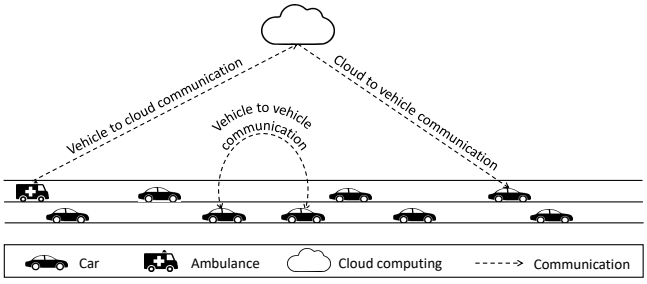

Figure 1: Intelligent Transportation System scenario

experimental results support our guess that formal methods are actually applicable in the performance analysis of smart CPS.

The paper is organized as follows. Section 2 reports our motivating example, Section 3 shows the performance modeling and experimental results. The validation with simulation is illustrated in Section 4. Related work is discussed in Section 5, and Section 6 concludes the paper.

2 RUNNING EXAMPLE

In this section we describe a smart CPS example selected from the Intelligent Transportation System (ITS) domain since it emphasizes dynamicity and adaptability to the changing environment. This type of environment is defined *intelligent* because it is supported by some form of intelligence, i.e., control logic added to the system itself or to its components, in order to optimize its overall quality, achieve particular goals in certain circumstances or fulfill specific functional and extra-functional requirements.

Figure 1 illustrates an ITS scenario for a particular road section. For the sake of simplicity, similarly to [30], our case study considers a freeway segment without on-ramps and off-ramps sections. The system comprises *normal vehicles* (i.e., cars), *special vehicles* (i.e., ambulances), a wireless communication network, and a remote *cloud* computing infrastructure. The entities are able to communicate with each other by exchanging data (see the arrows) using different communication media (e.g., 5G network) and different strategies (e.g., vehicle to vehicle communication, vehicle to cloud communication). All the vehicles are autonomous, they can sense the environment (i.e., messages sent by other vehicles) and be programmed to move without human intervention.

The system requirement we consider is to minimize the travel time of special vehicles crossing the road section. More specifically, the ITS should satisfy the following global requirement:

req0: "*The system shall maximize the traffic flow that allows special vehicles to travel the road (by encountering as less traffic as possible) and reach their destination soon*".

This requirement has to be fulfilled in any road configuration and traffic flow conditions until congestion. In fact, after road saturation, a travel time analysis is pointless as it is impossible for special vehicles to find space and overcome other vehicles.

Starting from **req0**, to assess the performance of the system, another requirement is derived, specifically **req1**: "*Special vehicles shall not be hindered by regular vehicles*"

We are interested in analyzing the traffic flow dynamics and congestion points in different road configurations. The analysis is focused on how the behavior of regular vehicles affects special vehicles in different traffic conditions.

To this end, a new requirement (sub-requirement of **req1**) is defined for the travel time of special vehicles, i.e., **req1.1**: *The average Travel Time for Special vehicles (TTS) has to be:* $lower\_bound \leq TTS \leq optimum + (optimum * 50\%)$. This expresses that the travel time for special vehicles must not be less than its lower bound, defined in terms of real world constraints, i.e., regulated by the physic law: vehicles with a specific speed can not have a travel time less than $lower\_bound = \frac{roadlength}{speed}$, i.e., the optimal value.

Moreover, in order to be acceptable, the travel time for special vehicles can suffer of a delay that is at maximum equal to the 50% of the optimal time. This requirement is derived by observing at the relation between the travel time and traffic delay, as a function of the flow rate, for generating unsaturated traffic conditions [26].

In our previous work [4] we analyzed a smart parking application where cars need to communicate with hot-spots to find an empty spot. However, the ITS domain entails a set of dynamic requirements (i.e., ambulances show up in an unpredictable way) that allow us to consider the system evolution and extend our analysis framework to runtime adaptation concerns.

3 PERFORMANCE-BASED MODELING AND ANALYSIS

For the requirements verification, depending on the domain and other aspects of the system under study, different techniques can be used in this phase of performance-based modeling and analysis. Consider as examples mathematical models (e.g., differential equations), performance models (e.g., markovian models, petri nets) and many others. In this paper we focus on state-of-the-art formal performance models, namely Queueing Networks (QN) [18]. Goal of this study is to understand (*i*) the modeling power of this formal method, and (*ii*) to what extent the obtained results can help the system engineering in making informed design decisions.

In the literature there are several works on traffic flow modeling using formal analytical techniques, but they often rely on knowing the actual traffic behavior [36]. From the data collected by monitoring the traffic, it is possible to derive the probability distribution or breakdown points of the traffic flow that identify if the system is congested or not [37]. Moreover, these formal analytical techniques are able to analyze the traffic from a macroscopic point of view, i.e., considering all the agents equal and looking at the traffic flow only. This means that there is no differentiation in the type of vehicles (e.g., normal or special), in their behavior (e.g., only specific type of vehicles can perform some actions like overtake another vehicle) as well as communication capabilities. Because of these limitations, to verify the given requirements, microscopic traffic flow models are needed for our analysis [12].

To conduct a microscopic investigation, our analysis relies on a queueing network model based on the principles of the cellular automata for traffic modeling [36]. In the following we describe the formalism adopted and then its application to the running example (see Section 2).

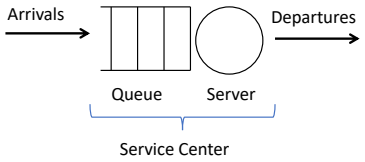

Figure 2: Single service center queueng model

3.1 Queueing network models

A queueing network (QN) model is a collection of interacting *service centers* representing system resources and a set of *jobs* representing the users sharing the resources [18]. Service centers model system resources that process customer request. Each service center is composed of a *Server* and a *Queue*. Figure 2 illustrates a simple QN model with a single service center highlighting its main components. Queues can be characterized by a finite or an infinite length. Service centers are connected through *Links* that form the network topology. Servers process *jobs* retrieved from their queue following a specific policy (e.g., FIFO). Each processed request is then routed to another service center through connections provided by links. More precisely, each server, contained in every service center, picks the next job from its queue (if not empty), processes it, and selects one link that routes the processed request to the queue of another service center. It is possible to specify a policy for link selection (e.g., probabilistic, round robin, etc.). The time spent in every server by each request is modeled by exponential distributions. Jobs are generated by source nodes connected with links to the rest of the QN. Delay centers are nodes of the QN connected with links to the rest of the network exactly as service centers, but they do not have an associated queue. Delay centers are described only by a service time, with a continuous distribution. They correspond to service centers with infinite servers. In other words, QN representation is a direct graph whose nodes are service centers and their connections are represented by the graph edges. Jobs go through the graph's edge set on the basis of the behavior of customers' service requests.

After modeling a system as a QN, the model has to be evaluated in order to determine quantitative performance metrics, such as:

- *Utilization*: the ratio between the server's busy time over the total time;
- *Response Time*: the interval between the submission of a request into the QN and output of results;
- *Queue Length*: the average queue length for a given service center;
- *Throughput*: the number of requests processed per unit of time (UT).

The above measures are defined for a single service center, but they can also be applied to the whole network, thus to get system evaluations. Several techniques can be used to evaluate these indices, from bounds analysis to simulation [18]. In the following we describe the techniques adopted in our running example.

3.2 QN for the performance modelling of ITS

To model the ITS, we consider the road as a grid where each cell represents a portion of the road which, in the base case, is entirely occupied by a vehicle. Each vehicle moves with a speed abstracted

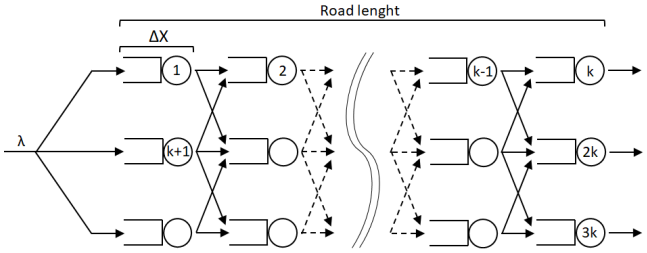

Figure 3: Queueing network architectural model of a road

in *number of cells per time unit*; it can change lane if there is more than one cell, and only if there is enough safe space to do so. Its representation with a QN is illustrated in Figure 3. The road is modeled with a chain of queuing centers. Each one of these service centers represents a piece of road with a specific length ΔX. Our assumption is that queuing and serving centers are jointly considered and represent a unique cell of the grid that can contain only one vehicle at a time (see ΔX in Figure 3).

After defining the length of a road section, k queuing centers are needed to model a road of length *road length* where k is obtained as $\lceil \frac{road\ length}{\Delta X} \rceil$. The chain of k elements models a road lane where a vehicle can go from a road section only to the next one (we exclude the case of a vehicle driving in the opposite wrong way). Each vehicle belongs to a type that specifies which is the average speed of the vehicle during the drive. Each service station has a service time based on the speed of the vehicle it is serving w.r.t. ΔX.

Moreover, to include some oscillation effects on the speed due to the human factor, each service time is calculated as exponentially distributed with a mean equal to the vehicle segment travel time that we remind to be $\frac{vehicle\ speed}{\Delta X}$. On average, without obstacles, a vehicle will cross the road in its optimal time. To model a multiple-lane road, we first modeled a single lane road, than we put other chains next to the first one and inserted a new link between queuing nodes of adjacent lanes, see Figure 3. With this model, a vehicle can advance straight forward in the next piece of road or go randomly to the other lane.

The vehicles flow is arriving into the system with some rate λ that is randomly routed to one of the lanes. In this configuration, vehicles go randomly along each available spot along the lanes. As specified by the requirements, different classes of vehicles are analyzed while crossing a particular road showing arbitrary length and a certain number of lanes. To solve this model and obtain the performance indices of interest, we apply the Mean Value Analysis (MVA) algorithm [18], which is a simple yet powerful technique allowing the exact computation of performance indices when the model satisfy given constraints. In this experiment we adopted the `queueing` package[2] to verify the fulfillment of requirement *req1*.

In the considered scenario, the number of ambulances is the 5% of the whole traffic. Besides, we tested a 2 lanes road with three different road lengths: 1600, 2400, and 3200 meters. We considered a ΔX of 7.5 meters for each cell of the grid, hence to cover 1600 meters of road we used 214 queuing centers for each lane, 321 centers for 2400 meters and 428 for 3200 meters. The applied average vehicle speed is 72 km/h for the car and 140 km/h for the ambulances.

[2]https://www.moreno.marzolla.name/software/queueing/

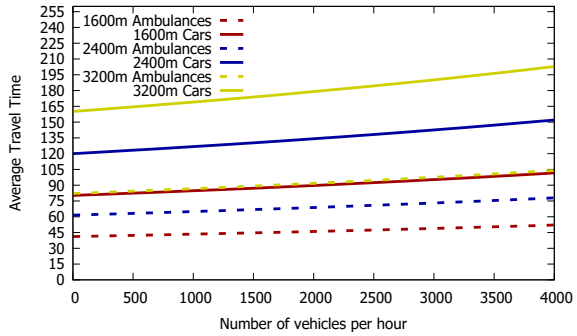

Figure 4: MVA results of traffic cars flow with overtake

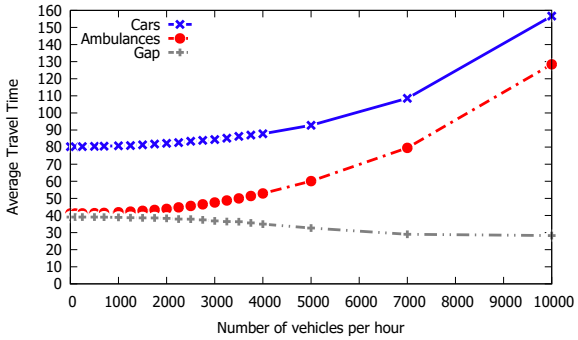

Figure 5: QN simulation with overtake in 2 lanes

According to the results of this scenario, reported in Figure 4, *req0* is partially satisfied, since no congestion is detected with this analysis, and the travel time requirement *req1.1* is validated. In fact, for each experiment, the ambulances require always less than 40*sec* per kilometer on average to cross the road section. Hence, the average travel time never exceeds 50% of the optimal value, which can be easily calculated by considering the average speed of the vehicles and the road length. Moreover, not surprisingly, by increasing the road length, the average travel time of both normal vehicles and ambulances linearly increase.

As said above, this preliminary analysis suggests that the requirement *req1.1* is satisfied. However, we can notice that in the graphs of Figure 4, the average travel time of ambulances linearly increases with the traffic flow. This behavior is suspicious because the expectation was to see that the average travel time of the ambulances converges to the one of the vehicles for high traffic densities (e.g., 4k vehicles per hour). This could entail that the model is not enough representative of the domain under investigation or that the converging trend does not result from the number of vehicles per hour tested for the system.

To evaluate the power of the adopted formalism, we try to improve the QN model with a more realistic overtake behavior and then perform the verification with configurations up to 10k vehicles. Indeed, with the random routing introduced in the previous model, the vehicles have a higher probability to find some car slower in front when the flow increases. This model is then refined with a smarter routing rule: the *Join the Shortest Queue* (JSQ), i.e., a vehicle that finds a slower car in front of it can chose to move in one empty adjacent queue, if any. The same rule is adopted for the routing of new vehicles. With these new enhancements, the assumptions underlying MVA do not hold any more because these features are state dependent.

The model is solved by means of the JMT tool [7], and results are reported in Figure 5. As expected, a smart routing rule improves the average travel time of cars. Moreover, the average travel time of the ambulances asymptotically reaches the average travel time of the cars with an increase of traffic flow, because the ambulance is not able to perform overtakes with a fully congested road. However, we expected to find two break points between low and high traffic jams: (*i*) when the traffic density starts affecting the ambulance travel time, and (*ii*) when the ambulance travel time becomes asymptotic to the cars travel time, due to the high traffic jam. Unfortunately, the graph depicted in Figure 5 does not show break points or an asymptotic trend in the ambulances travel time's curve. Besides, there is a gap between the two travel times that requires further investigations. To verify these hypotheses we simulate the ITS and compare the obtained results.

4 MODEL VALIDATION THROUGH SIMULATION

In this section we compare the results obtained from QN formal models (see Section 3) with the ones derived through simulation. This allows to validate the microscopic traffic flow model designed by means of QN.

To this end, we implemented the ITS case study in MovSim [34], i.e., a multi-model Java open-source lane-based microscopic traffic simulator. According to previous models, the key aspects of our system are: the road and the behavior of different types of vehicles, namely ambulances and cars. The road is configured with a certain number of lanes, whereas the behavior of vehicles is defined according to the intelligent driver model [15] and the lane-changing model [14] already implemented in the MovSim.

Analogously to the previous analysis, the road is a straight section with two lanes. The average speed of normal vehicles crossing the road is 70 km/h. The average speed of ambulances crossing the road is 140 km/h. The number of ambulances crossing the road section is equal to 5% of the whole traffic. The experiments run 3 hours of simulated time for configurations up to 4k vehicles per hour. This means that simulations deal with scenarios including up to 12k communicating vehicles.

Similarly to QN analysis (see Section 3), our experimentation first aims at assessing the behavior of the system by increasing the length of a 2 lanes road. This results in 3 simulations with different road lengths. The road of the first experiment is 1600 meters long, the road of the second is 2400 meters, while the last one is 3200 meters long, see Figure 7. With respect to the results obtained with the traffic model introduced in the design phase, we still can observe that increasing the road length causes a linear increase of the average travel time of the cars (see Figure 7).

We notice here an important difference between the two sets of results. In particular, we can observe that the average travel time of the ambulances converges to the average travel time of normal cars starting from 4k vehicles per hour. This is a behavior that is inconsistent with the results gathered in the previous analysis

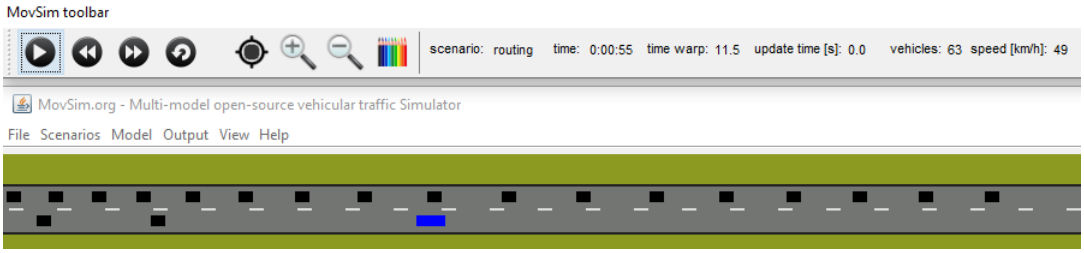

Figure 6: ITS Simulation with MoveSim

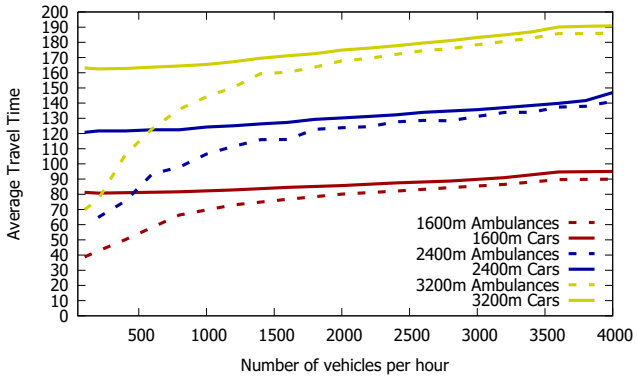

Figure 7: Simulation results

and that is worth investigating. In fact, in the QN analysis, such convergence behavior results only after 10k vehicles per hour. Requirement *req1.1* is not satisfied any more for any road lengths of 2 lanes when there are more than 1k vehicles per hour, see Figure 7.

Discussion. To summarize, even if the results reported in Figure 5 are logically correct, they are far to be a precise description of the actual environment. This is due to the model itself, its nature and the design assumptions that have been made to build a high-level model of vehicles traffic flow. The policy of JSQ is a rough emulation of an overtake policy that takes into consideration the state of the queue when looking for a free spot. However, this policy does not guarantee safety in case a vehicle decides to overtake another one. This leads to a situation where any vehicle is able to overtake in any traffic condition, also in a fully congested road. We deduce that this is the reason why the curves of Figure 5 show no evident break point or asymptotic behavior.

To capture these aspects, it is necessary to consider different formalisms (while taking into account further properties like scalability and representativeness) or to build more realistic simulations. Although this demonstrates the limitations of analytical formalisms when the system complexity grows, the QN-based analysis is still useful because it provides an understanding of the system behavior. Predictive results (obtained through MVA or boundary analysis) can be considered base lines for further analyses and comparisons. It is worth to remark that MVA is very efficient, results are produced in order of seconds, whereas simulations are computationally much more expensive, we experienced an average running time between 5 and 20 minutes for our experiments.

5 RELATED WORK

Formal methods for the analysis of systems performance requirements have been successfully applied in the past to several domains (e.g., [7]). However, for the majority of modern complex systems obtaining satisfactory and omni-comprehensive formal models (and analyses) could be quite challenging. Therefore the application of formal methods is often suggested at design time, when the software engineer has to verify a complex system with respect to a set of requirements, and there is often no need to consider the precise structure of the system and the details of its elements [20].

When the Quality of Service (QoS) requirements are not tied to the concrete behavior/execution of the system, high-level formal QoS models can be selected to preliminarily assess the designed system. Analytical models can be adopted in this phase depending on the specific domain of the system under investigation and the type of QoS requirements to validate. A large body of analysis techniques have been used in the literature to deal with QoS-based validation – e.g., well-known analytical QoS models adopted at design-time are Queuing Networks [16], Petri Nets [21] and Stochastic Automata Networks [33]. When analyzing QoS characteristics of software systems, many approaches have been proposed to optimize different quality indicators [1]. The main quality attributes that have been evaluated in literature are: performance [27], cost [5], reliability [3], availability [11], but also trade-off analysis among multiple quality attributes has been pursued [8].

When the system exhibits complex behaviors, simulation-based analysis techniques are usually applied to collect results that are otherwise impossible to gather. Some examples can be found in [6, 24]. Simulation-based approaches for systems' analysis often use co-simulation based techniques to deal with the inherent complexity of hardware-software system coexistence [25]. Examples of systems where co-simulation is used range from smart grids [10] to wireless networked control systems [2]. In such cases, the complexity of these applications makes the co-simulation a necessary tool during system development.

There are alternative approaches to precisely analyze the properties of CPS. For instance, mathematical models of the system can be built, and a formal verification of its behaviour can be performed [22]. Two methods are used, generally: (i) model checking [13] and (ii) theorem proving applied to a high-order logic description of the system [29]. This way, theorem proving may be adopted to perform a formal analysis of the system properties. These approaches can be considered complementary to ours, and it is part of our future research to understand the synergies among them, if any.

6 CONCLUSIONS

In this paper we have presented an exploratory study to investigate the applicability of formal models like QN for the performance analysis of smart CPS. To this end, we have considered an intelligent transport system as case study and we have compared the results obtained applying a general QN model and a more specific simulation model. Experimental preliminary results obtained with this exploratory study highlight some key aspects: (*i*) QN analysis, despite its limitations, can be useful for a first understanding of the system behavior, and can represent a baseline for further analyses and comparisons; (*ii*) some congestion behaviors are not easily captured by QN models and therefore these models can give significant insights only for a limited set of aspects; (*iii*) the need of more powerful models able to capture the whole system behavior and to provide performance results in short time.

As future work, we intend to investigate multi-modeling approaches able to deal with different aspects of the system in a combined way. From a more general view point, independently of the formalism adopted for the performance models, there is the need of an integrated approach that allows the automatic derivation of performance models starting from an unambiguous specification of the system behavior.

ACKNOWLEDGMENT

This research has been partially supported by the Swedish Knowledge Foundation, Grants No. 20150088 and No. 20170232, and the MIUR PRIN project titled "Designing Spatially Distributed Cyber-Physical Systems under Uncertainty (SEDUCE)".

REFERENCES

[1] Aldeida Aleti, Barbora Buhnova, Lars Grunske, Anne Koziolek, and Indika Meedeniya. 2013. Software Architecture Optimization Methods: A Systematic Literature Review. *IEEE Trans. Software Eng.* 39, 5 (2013), 658–683.

[2] Martin Andersson, Dan Henriksson, Anton Cervin, and K Arzen. 2005. Simulation of wireless networked control systems. In *European Control Conference on Decision and Control*. 476–481.

[3] Asoke Kumar Bhunia, Laxminarayan Sahoo, and Dilip Roy. 2010. Reliability stochastic optimization for a series system with interval component reliability via genetic algorithm. *Appl. Math. Comput.* 216, 3 (2010), 929–939.

[4] Tomás Bures, Vladimír Matena, Raffaela Mirandola, Lorenzo Pagliari, and Catia Trubiani. 2018. Performance Modelling of Smart Cyber-Physical Systems. In *International Conference on Performance Engineering*. 37–40.

[5] Lei Cao, Jian Cao, and Minglu Li. 2005. Genetic algorithm utilized in cost-reduction driven web service selection. In *International Conference on Computational and Information Science*. 679–686.

[6] Valeria Cardellini, Emiliano Casalicchio, Vincenzo Grassi, Stefano Iannucci, Francesco Lo Presti, and Raffaela Mirandola. 2017. MOSES: A Platform for Eperimenting with QoS-Driven Self-Adaptation Policies for Service Oriented Systems. In *Assurances on Software Engineering for Self-Adaptive Systems III*. Springer, 409–433.

[7] Giuliano Casale and Giuseppe Serazzi. 2011. Quantitative system evaluation with Java modeling tools. In *International Conference on Performance Engineering*. 449–454.

[8] Atakan Doğan and Füsun Özgüner. 2005. Biobjective scheduling algorithms for execution time–reliability trade-off in heterogeneous computing systems. *Comput. J.* 48, 3 (2005), 300–314.

[9] Allan Edgard Silva Freitas and Romildo Martins da Silva Bezerra. 2016. Performance Evaluation of Cyber-Physical Systems. *ICIC Express Letters* 10 (2016).

[10] Tim Godfrey, Sara Mullen, David W Griffith, Nada Golmie, Roger C Dugan, and Craig Rodine. 2010. Modeling smart grid applications with co-simulation. In *International Conference on Smart Grid Communications*. 291–296.

[11] Huipeng Guo, Jinpeng Huai, Huan Li, Ting Deng, Yang Li, and Zongxia Du. 2007. Angel: Optimal configuration for high available service composition. In *International Conference on Web Services*. 280–287.

[12] J. Harri, F. Filali, and C. Bonnet. 2009. Mobility models for vehicular ad hoc networks: a survey and taxonomy. *IEEE Communications Surveys Tutorials* 11, 4 (2009), 19–41.

[13] Kenan Kalajdzic, Cyrille Jégourel, Anna Lukina, Ezio Bartocci, Axel Legay, Scott A. Smolka, and Radu Grosu. 2016. Feedback Control for Statistical Model Checking of Cyber-Physical Systems. In *International Symposium on Leveraging Applications of Formal Methods, Verification and Validation*. 46–61.

[14] Arne Kesting, Martin Treiber, and Dirk Helbing. 2007. General Lane-Changing Model MOBIL for Car-Following Models. *Transportation Research Record* 1999, 1 (2007), 86–94.

[15] Arne Kesting, Martin Treiber, and Dirk Helbing. 2010. Enhanced Intelligent Driver Model to Access the Impact of Driving Strategies on Traffic Capacity. *Philosophical transactions. Series A, Mathematical, physical, and engineering sciences* 368 (2010), 4585–605.

[16] P. Kuehn. 1979. Approximate Analysis of General Queuing Networks by Decomposition. *IEEE Transactions on Communications* 27, 1 (1979), 113–126. https://doi.org/10.1109/TCOM.1979.1094270

[17] Peter Gorm Larsen, John Fitzgerald, Jim Woodcock, René Nilsson, Carl Gamble, and Simon Foster. 2016. Towards semantically integrated models and tools for cyber-physical systems design. In *International Symposium on Leveraging Applications of Formal Methods*. Springer, 171–186.

[18] E. D. Lazowska, J. Zahorjan, G. Scott Graham, and K. C. Sevcik. 1984. *Computer System Analysis Using Queueing Network Models*. Prentice-Hall.

[19] Jay Lee, Behrad Bagheri, and Hung-An Kao. 2015. A cyber-physical systems architecture for industry 4.0-based manufacturing systems. *Manufacturing Letters* 3 (2015), 18–23.

[20] Massimo Marchiori. 1998. Light Analysis of Complex Systems. In *ACM Symposium on Applied Computing*. ACM, New York, NY, USA, 18–22.

[21] T. Murata. 1989. Petri nets: Properties, analysis and applications. *Proc. IEEE* 77, 4 (1989), 541–580.

[22] Pierluigi Nuzzo, Jiwei Li, Alberto L. Sangiovanni-Vincentelli, Yugeng Xi, and Dewei Li. 2019. Stochastic Assume-Guarantee Contracts for Cyber-Physical System Design. *ACM Trans. Embedded Comput. Syst.* 18, 1 (2019), 2:1–2:26.

[23] Lorenzo Pagliari, Raffaela Mirandola, and Catia Trubiani. 2017. Multi-modeling Approach to Performance Engineering of Cyber-Physical Systems Design. In *International Conference on Engineering of Complex Computer Systems*. 142–145.

[24] Colin Paterson and Radu Calinescu. 2017. Accurate Analysis of Quality Properties of Software with Observation-Based Markov Chain Refinement. In *International Conference on Software Architecture*. 121–130.

[25] James A Rowson. 1994. Hardware/Software Co-Simulation. In *International Conference on Design Automation*, Vol. 94. 6–10.

[26] Hemant Kumar Sharma, Mansha Swami, and BajrangLal Swami. 2012. Speed-flow analysis for interrupted oversaturated traffic flow with heterogeneous structure for urban roads. *International Journal for Traffic and Transport Engineering* 2, 2 (2012), 142–152.

[27] Vibhu Saujanya Sharma and Pankaj Jalote. 2008. Deploying software components for performance. In *International Symposium on Component-Based Software Engineering*. 32–47.

[28] Jianhua Shi, Jiafu Wan, Hehua Yan, and Hui Suo. 2011. A survey of Cyber-Physical Systems. In *International Conference on Wireless Communications & Signal Processing WCSP*. 1–6.

[29] Yasser Shoukry, Pierluigi Nuzzo, Alberto Puggelli, Alberto L. Sangiovanni-Vincentelli, Sanjit A. Seshia, and Paulo Tabuada. 2017. Secure State Estimation for Cyber-Physical Systems Under Sensor Attacks: A Satisfiability Modulo Theory Approach. *IEEE Trans. Automat. Contr.* 62, 10 (2017), 4917–4932.

[30] Kateřina Staňková and Bart De Schutter. 2010. On freeway traffic density estimation for a jump Markov linear model based on Daganzo's cell transmission model. In *Proceedings of the International Conference on Intelligent Transportation Systems*. 13–18.

[31] Jonette M Stecklein, Jim Dabney, Brandon Dick, Bill Haskins, Randy Lovell, and Gregory Moroney. 2004. Error cost escalation through the project life cycle. *NASA Technical Report* (2004).

[32] Muhammad Umer Tariq, Jacques Florence, and Marilyn Wolf. 2014. Design Specification of Cyber-Physical Systems: Towards a Domain-Specific Modeling Language based on Simulink, Eclipse Modeling Framework, and Giotto.. In *ACESMB@ MoDELS*. 6–15.

[33] M. A. L. Thathachar and P. S. Sastry. 2002. Varieties of learning automata: an overview. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)* 32, 6 (2002), 711–722. https://doi.org/10.1109/TSMCB.2002.1049606

[34] M. Treiber and A. Kesting. 2010. An Open-Source Microscopic Traffic Simulator. *IEEE Intelligent Transportation Systems Magazine* 2, 3 (2010), 6–13.

[35] A. S. Vincentelli. 2015. Let's get physical: Adding physical dimensions to cyber systems. In *IEEE/ACM International Symposium on Low Power Electronics and Design (ISLPED)*. 1–2.

[36] Lanhang Ye and Toshiyuki Yamamoto. 2018. Modeling connected and autonomous vehicles in heterogeneous traffic flow. *Physica A: Statistical Mechanics and its Applications* 490 (2018), 269–277.

[37] Jiyoun Yeon, Lily Elefteriadou, and Siriphong Lawphongpanich. 2008. Travel time estimation on a freeway using Discrete Time Markov Chains. *Transportation Research Part B: Methodological* 42, 4 (2008), 325–338.