Operational Validation of Simulation Runs on Automated Simulation Platforms

Abstract - *Scenario based testing is one of the major concepts on the route towards establishing a process for homologation of autonomous vehicles. While there is already a lot of progress in the domains of scenario exploration and standardization of interfaces for simulation models enabling automated scenario simulation the question of valid scalability is yet to be answered. Although container orchestration systems are widely used throughout several industries, the impact of scheduling algorithms on simulation run validity is not studied yet. Operational validity can become a concern as the hardware abstraction of such orchestration systems may oppose the requirement to test software on hardware that resembles the production system. This paper presents a concept on how to formalize operational validity and use this formalization to prove plausible simulation execution.*

Keywords: *validity, simulation, cloud computing, orchestration*

1. Introduction

With the rising complexity of automotive systems the effort to test and verify safety is also increasing (Koopman and Wagner, 2016; Li, et al., 2016). To get new and complex features certified for the public road they have to work for a certain amount of driven kilometers without failures (Nidhi Kalra and Susan M. Paddock, 2016; Schilling and Schultz, 2016). Since the testing effort will only rise accordingly with more features the industry evaluates how much can be done by simulating these distances. Once this is known the acceleration of the test process will be achieved by massive automation and parallelization of simulation runs. A simulation run consists of heterogeneous programs simulating different domains to eventually prove the safe passage of an automotive system through different critical scenarios. Transparency in how these test results are produced is important since they are a significant part of the argument for safety of complex automotive systems. This includes the information if a simulation run was conducted with sufficient hardware capacity. For systems-under-test (SUT) like autonomous driving functions, which may rely on correctly timed communication of their components, the availability of sufficient hardware capacity in simulation has direct impact on the validity of the results.

Validation of a simulation model or a complete simulation run has multiple aspects, one of them being the operational validation. According to Sargent, Goldsman, and Yaacoub, 2016, ”Operational validation is determining whether the simulation model’s output behavior has the accuracy required for the model’s intended purpose over the domain of the model’s intended applicability”. Assuring operational validity for individual simulation models is done by comparing simulated data with measured data of a physical system (Sargent, 2020; Sargent, Goldsman, and Yaacoub, 2016). As vehicle systems are transitioning from several control units to centralized computing platforms best practices used in general software development will be used more in the development of simulation models and whole vehicle system software - which includes more test automation (Traub, Maier, and Barbehön, 2017). Taking test automation systems and cloud and edge computing into the equation might have a significant impact on the validity of (real-time) simulation. Assembling operationally valid simulation models does not guarantee the operational validity of the assembly, as valid unit tests do not guarantee the correct behavior of the integrated system (Ammann and Offutt, 2016; Bélanger and Venne, 2010). As the operational environment of a simulation e.g., driving agents for other road users also becomes more complex it is necessary to evaluate the complete simulation configuration and its respective execution or simulation run. Test scheduling and orchestration within cloud and edge systems may lead to simulation runs interfering with each other if they run on the same hardware, for example disrupt the timing of signals (“IEEE Standard for Modeling and Simulation (M & S) High Level Architecture (HLA)– Framework and Rules” 2010). Due to the increasing agility in the development of software for vehicle systems the SUTs also do not necessarily have the maturity of an optimized product when they enter simulation (Collins, Dias-Neto, and Lucena Jr., 2012).

2. Research Questions

Operational validity asks for accuracy confirmation in the intended application domain. For most simulation models this accuracy is correlated with the time spent calculating results e.g., trajectories for a vehicle controller or accurate virtual laser rays for a LiDAR sensor model. This can become a problem with real-time simulations on overloaded computing systems. In real-time simulation the scheduling and communication has to be similar to the production system in the vehicle. Even under the assumption that a fraction of all needed simulations have to be done in real-time, due to faster development cycles it would be beneficial for the OEM to leverage the possibilities of cloud computing techniques. It is plausible to assume that throughout a scenario the controlling software of a vehicle has different hardware requirements to work correctly. This fluctuation can make it difficult to estimate the upper bounds of the needed hardware capacity. These upper bounds are necessary to know to ensure the availability of the required hardware capacity at scheduling time of the simulation on a cloud platform. Estimating too generously would result in efficiency loss and estimating too frugally might result in invalid simulation results. The latter case is potentially invalid as processes might starve each other for computing resources and consequentially disturb the timings and calculation accuracy of the involved SUTs.

This leads to the following research questions:

- How can the operational validity of an assembled SUT be formalized?
- When is a real-time simulation run operationally valid?
- What is a scheduling strategy that is aware of the operational validity of real-time simulation runs?

3. Methodology

The following subsections will describe how the respective research questions will be answered, the overall approach can be seen in figure 1. As this approach is a concept for future research an illustrating example will be used to make it more approachable.

3.1. Operational Requirements of Scenarios

To build a scheduler that is aware of the operational requirements of a simulation model assembly it is necessary to have an initial estimate of said requirements (see figure 1 'Baseline Preparation'). These estimates are done by measuring the hardware load when simulating different types of scenarios (e.g. highway, urban or country roads). It is plausible to assume that the hardware requirements of a SUT vary with the scenario type they are confronted with one example being that higher simulated velocities need a higher update rate to simulate correctly in real-time. Furthermore, it is likely that a complex assembly e.g., autonomous driving functionality uses some form of best-effort computation as the production system will be a closed system with known components so the hardware usage can be optimized. To test this hypothesis a representative scenario will be created for each scenario type. An openly available driving function (Heß, 2020) will be simulated in these scenarios and evaluated for the impact of different metrics (e.g. hardware usage metrics in conjunction with other validity metrics). These measurements will be done on an idle system and a busy system with different deployment techniques. The collected data will be evaluated in order to chose metrics to include into a statistically sophisticated simulation run benchmark (see subsection 3.2).

Example

Suppose the SUT is known by experience to be working as intended in urban scenarios on a dedicated machine with 8 GB of RAM. In this example suppose the scenario specifies a turn to the left on an intersection with oncoming traffic. "Working as intended", meaning there is no collision with the oncoming traffic and the trajectory of the SUT stays similar over reiterations of this simulation. To further specify this information the SUT will be simulated in a reference scenario for urban traffic whilst a defined portion the machine's RAM is blocked by another process. Afterwards the trajectories of both experiments (blocked RAM and available RAM) will be compared to test the stated hypothesis. In reality more metrics will be recorded and also other forms of hardware blocking will be compared e.g., occupied processor cores.

3.2. Simulation Run Benchmarking

Once it is known which metrics have impact on the performance of the SUT the next step is to determine the quantitative variation. This will be done by measuring scenario exploration of the same type with different criticality to get a tolerance range for the scenario type. This range will be the estimate needed for the scheduling (see first artifact in Figure 1).

Example

Suppose the measurements resulted in a significantly different driving trajectory once the available RAM was below 4 GB and the simulation produced crashes or other faulty behavior once it was below 3 GB. Now there are a number of concrete scenarios, for example 100, created from the same logical scenario (urban intersection). The SUT will be simulated in all 100 scenarios on a dedicated system while all identified hardware metrics are recorded. The measurements are statistically evaluated to determine the range in between the hardware metrics can be considered as 'working as intended'. The upper bound of this range plus a buffer will be the minimum capacity needed for a valid result from this SUT in this type of scenario. These ranges will determined for different classes of scenarios and attached to the SUT.

3.3. Operational Validity-Aware Scheduling

With the a-priori information available the scheduling method depicted in figure 1, 'Operational Validity Aware Scheduling', can be implemented. Operational validity of the respective simulation runs of a simulation campaign will be assured through the automated comparison of a-priori and a-posteriori information and if necessary re-simulation on a less busy system.

Example

The calculated ranges are baselines for this particular SUT and will be used for automatic scheduling. Furthermore they provide a 'window of operational validity' which can be checked after every simulation on a busy computation cluster. If the simulation had less hardware resources available this might indicate the scenario was not valid. In that case the specific simulation is repeated on a dedicated cluster where no other process can interfere.

4. Discussion

It is important to keep actual technological solutions in mind when transferring methods to other domains.

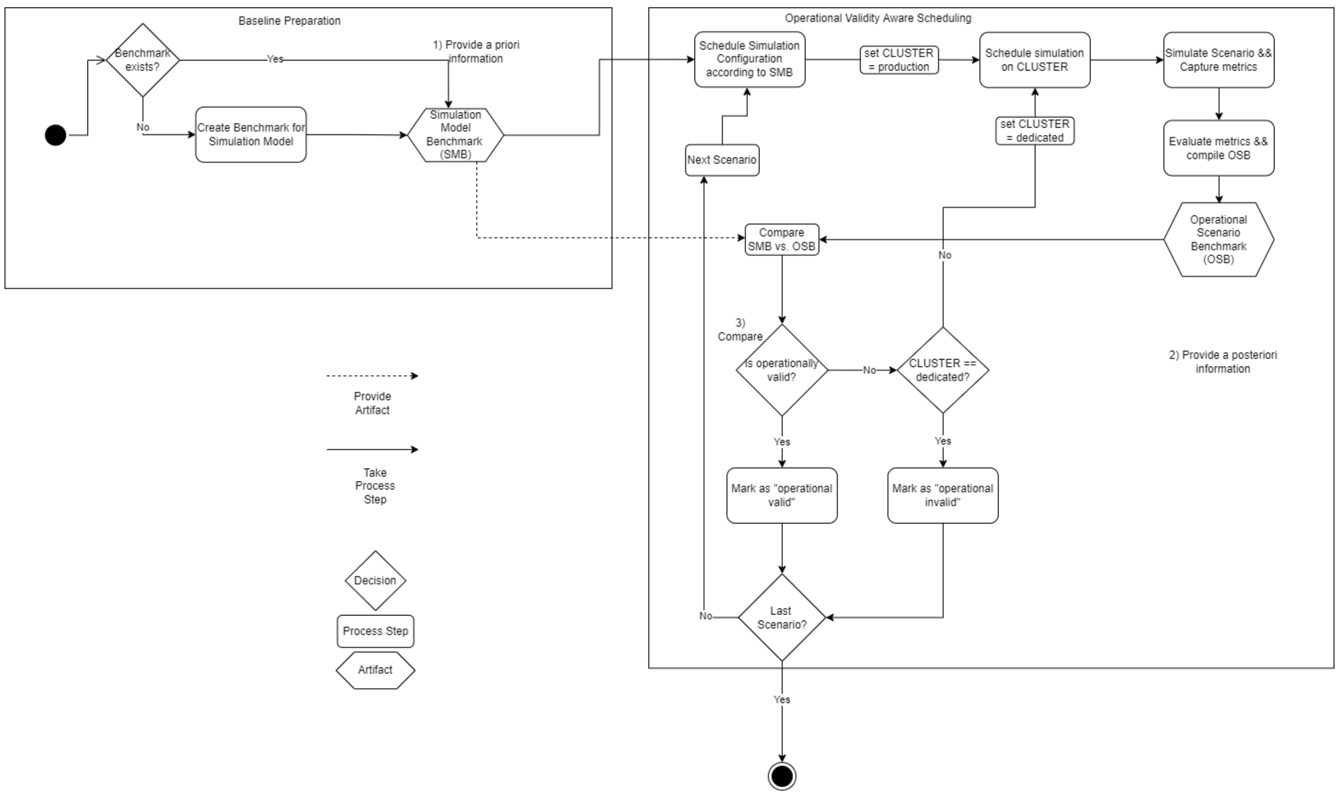
Baseline Preparation
Benchmark exists?
Yes
No
1) Provide a priori information
Create Benchmark for Simulation Model
Simulation Model Benchmark (SMB)
Operational Validity Aware Scheduling
Schedule Simulation Configuration according to SMB
set CLUSTER = production
Schedule simulation on CLUSTER
Simulate Scenario && Capture metrics
Evaluate metrics && compile OSB
Operational Scenario Benchmark (OSB)
set CLUSTER = dedicated
Next Scenario
Compare SMB vs. OSB
3) Compare
Is operationally valid?
CLUSTER == dedicated?
2) Provide a posteriori information
Mark as "operational valid"
Mark as "operational invalid"
Last Scenario?
Provide Artifact
Take Process Step
Decision
Process Step
Artifact

Figure 1: Workflow 'How to measure operational validity of simulation runs'

Cloud computing methods may introduce systemic errors to the simulation of a massive amount of scenarios. Stacking complex software on top of each other to automate processes that have been overseen by experts until now requires vigilance and the introspection into what these experts have been doing implicitly. With the proposed concept it would be possible to transform a portion of this domain knowledge into automatic tests. If this kind of empirical validation of operability becomes the norm for simulation result evaluation it would also promote reproducibility of simulation results.

5. Conclusion

This paper presents an approach for researching and evaluating the impact of applying cloud computing methods to real-time sensitive SUTs simulation. The next step will be the evaluation of the relationship of hardware metrics, scenarios and general simulation model validity.

References

Ammann, P. and Offutt, J., Dec. 1, 2016. Introduction To Software Testing. MAG ID: 1486172410.
Bélanger, J and Venne, P, 2010. The What, Where and Why of Real-Time Simulation. *Planet RT*, vol. 1, no. 1, pp. 25–29.
Collins, E., Dias-Neto, A., and Lucena Jr., V. F. d., July 2012. Strategies for Agile Software Testing Automation: An Industrial Experience. In: *2012 IEEE 36th Annual Computer Software and Applications Conference Workshops*. 2012 IEEE 36th IEEE Annual Computer Software and Applications Conference Workshops (COMPSACW). Izmir, Turkey: IEEE, pp. 440–445. ISBN: 978-1-4673-2714-5 978-0-7695-4758-9. `https://doi.org/10.1109/COMPSACW.2012.84`. Available at: ¡`http://ieeexplore.ieee.org/document/6341616/`¿ [Accessed Dec. 8, 2021].
Heß, D., Jan. 6, 2020. *Eclipse Automated Driving Open Research (ADORe)*. projects.eclipse.org. Available at: ¡`https://projects.eclipse.org/proposals/eclipse-automated-driving-open-research-adore`¿ [Accessed Dec. 20, 2021].
IEEE Standard for Modeling and Simulation (M & S) High Level Architecture (HLA)– Framework and Rules Aug. 2010. *IEEE Std 1516-2010 (Revision of IEEE Std 1516-2000)*. Conference Name: IEEE Std 1516-2010 (Revision of IEEE Std 1516-2000), pp. 1–38. `https://doi.org/10.1109/IEEESTD.2010.5553440`.
Koopman, P. and Wagner, M., 2016. Challenges in autonomous vehicle testing and validation. *SAE International Journal of Transportation Safety*, 4(1), pp. 15–24.
Li, L., Huang, W.-L., Liu, Y., Zheng, N.-N., and Wang, F.-Y., 2016. Intelligence testing for autonomous vehicles: A new approach. *IEEE Transactions on Intelligent Vehicles*, 1(2), pp. 158–166.
Nidhi Kalra and Susan M. Paddock, Dec. 1, 2016. Driving to Safety: How Many Miles of Driving Would It Take to Demonstrate Autonomous Vehicle Reliability? *Transportation Research Part A-policy and Practice*, 94. MAG ID: 2525936901, pp. 182–193. `https://doi.org/10.1016/j.tra.2016.09.010`.
Sargent, R. G., Dec. 14, 2020. Verification And Validation Of Simulation Models: An Advanced Tutorial. In: *2020 Winter Simulation Conference (WSC)*. 2020 Winter Simulation Conference (WSC). Orlando, FL, USA: IEEE, pp. 16–29. ISBN: 978-1-72819-499-8. `https://doi.org/10.1109/WSC48552.2020.9384052`. Available at: ¡`https://ieeexplore.ieee.org/document/9384052/`¿ [Accessed May 31, 2021].
Sargent, R. G., Goldsman, D. M., and Yaacoub, T., Dec. 2016. A tutorial on the operational validation of simulation models. In: *2016 Winter Simulation Conference (WSC)*. 2016 Winter Simulation Conference (WSC). ISSN: 1558-4305, pp. 163–177. `https://doi.org/10.1109/WSC.2016.7822087`.
Schilling, R. and Schultz, T., 2016. Validation of Automated Driving Functions. In: C. Gühmann, J. Riese, and K. von Rüden eds. *Simulation and Testing for Vehicle Technology*. Cham: Springer International Publishing, pp. 377–381. ISBN: 978-3-319-32344-

2 978-3-319-32345-9. https://doi.org/10.1007/978-3-319-32345-9_25. Available at: ¡http://link.springer.com/10.1007/978-3-319-32345-9_25¿ [Accessed Aug. 25, 2021].

Traub, M., Maier, A., and Barbehön, K. L., May 2017. Future Automotive Architecture and the Impact of IT Trends. *IEEE Software*, 34(3). Conference Name: IEEE Software, pp. 27–32. ISSN: 1937-4194. https://doi.org/10.1109/MS.2017.69.