Applicability of Model Checking for Verifying Spacecraft Operational Designs

Philipp Chrszon[1], Paulina Maurer[1], George Saleip[1], Sascha Müller[1], Philipp M. Fischer[1], Andreas Gerndt[2], and Michael Felderer[3]

Abstract: This is a summary of the paper *Applicability of Model Checking for Verifying Spacecraft Operational Designs* which has been published at the 26th International Conference on Model Driven Engineering Languages and Systems (MODELS 23).

Keywords: Aerospace, Formal Models, Formal Methods, Model Checking

The published version of this article is available at `https://doi.org/10.18420/modellierung2024_010`.

Summary

Guaranteeing safety and correctness is one of the main objectives in the development of space systems. This is challenging, since spacecraft consist of highly interconnected and interdependent parts which are developed by engineers of many different disciplines. Traditionally, validation and verification of space systems employ simulation and testing, where the system is run and observed for unintended behaviors. As full coverage of all possible runs is never achieved in practice, this can only show the presence of errors. In order to achieve full coverage, i.e., to prove the absence of errors, simulation is increasingly complemented by the application of formal methods, such as model checking. However, a major challenge for adopting model checking into the design process is its scalability. Usually, the whole state space of a system, which grows exponentially with, e.g., the number of parallel processes, must be explored. Dealing with this state-space explosion problem often requires expert knowledge. This severely hinders the widespread adoption of model checking in industry, including the aerospace domain [Da13].

We have investigated the applicability of model checking for the verification and analysis of spacecraft during early design phases. In particular, we examined up to which model size

[1] German Aerospace Center (DLR), Institute for Software Technology, Brunswik, Germany,
philipp.chrszon@dlr.de, https://orcid.org/0000-0002-8785-0272;
paulina.maurer@dlr.de, https://orcid.org/0000-0002-3954-6137;
george.nasralla@dlr.de, https://orcid.org/0000-0003-2123-3412;
sa.mueller@dlr.de, https://orcid.org/0000-0002-1913-1719;
philipp.fischer@dlr.de, https://orcid.org/0000-0003-2918-5195

[2] German Aerospace Center (DLR), Institute for Software Technology, Brunswik; University of Bremen, Bremen, Germany, andreas.gerndt@dlr.de, https://orcid.org/0000-0002-0409-8573

[3] German Aerospace Center (DLR), Institute for Software Technology, Cologne; University of Cologne, Cologne, Germany, michael.felderer@dlr.de, https://orcid.org/0000-0003-3818-4442

and complexity model checking is still tractable and fast enough to be incorporated into the design process. Our proposed verification approach targets the mode management of a spacecraft and is designed to rely on automated optimizations only, such that it may be used by non-experts in formal verification.

For modeling the mode management, we utilize a simplified variant of SysML state machines. An operational design may consist of one or more state machines corresponding to different subsystems or equipment, where transitions between modes may occur automatically or may be triggered by commands. Interactions and dependencies between state machines are expressed using mode constraints [Ei11]. We have defined the semantics of a set of state machines under certain mode constraints in terms of transition systems, a standard semantics for the formalization of operational behavior. Based on this semantics, we have implemented transformations from state machines into the modeling languages of several model-checking tools to enable a comparative evaluation. In particular, the model checkers SPIN, NuSMV, PRISM, and Storm have been selected as they support a wide range of fully-automatic techniques for mitigating the state-space explosion problem.

The experimental evaluation is based on a representative model that may arise within an early design phase of a satellite. The model is easily scalable by increasing the number of state machines, which in turn also increases the number of states exponentially. The model has been instantiated for a number of state machines ranging from 8 to 252 (ranging from 468 to 4×10^{70} states) and subsequently verified using the selected model checkers. The results show that checking for global and local deadlocks is possible within minutes or tens of minutes even for the largest instance. Furthermore, the peak memory consumption stayed below 4 GB for all tools and models. In conclusion, model checking is applicable for the selected use case, it can be used on commodity hardware, and is fast enough to be transparently integrated into the design process.

Data Availability

We provide a replication package [Ch23] comprising the generated models, measurements, scripts for running the experiments, and a Docker image that contains all required software.

References

[Ch23] Chrszon, P.: Applicability of Model Checking for Verifying Spacecraft Operational Designs - Artifact, version 1.2, Zenodo, 2023, doi: 10.5281/zenodo.8186567.

[Da13] Davis, J. A.; Clark, M. A.; Cofer, D. D.; Fifarek, A.; Hinchman, J.; Hoffman, J. A.; Hulbert, B. W.; Miller, S. P.; Wagner, L. G.: Study on the Barriers to the Industrial Adoption of Formal Methods. In (Pecheur, C.; Dierkes, M., eds.): Formal Methods for Industrial Critical Systems - 18th International Workshop, FMICS 2013, Madrid, Spain, September 23-24, 2013. Proceedings. Vol. 8187. Lecture Notes in Computer Science, Springer, pp. 63–77, 2013, doi: 10.1007/978-3-642-41010-9_5.

[Ei11] Eickhoff, J.: Onboard computers, onboard software and satellite operations: an introduction. Springer Science & Business Media, 2011.