The Role and Status of Model Checking

A Review of Literature and Survey of Some Uses for Model Checking in Society

Devin Gonier
Department of Computer Science
Lewis University
Romeoville, IL
DevinCGonier@lewisu.edu

***Abstract*—Model Checking serves as a primary method for formal verification in safety critical systems. This paper outlines some of the specific applications model checkers can have in various industries. Surveys done by others about model checkers in these industries are reviewed, and the railway industry is explored as an example. The railway industry is selected based on its growing significance, rapid changes, and the safety-critical nature and complexity of related software.**

Keywords—Model Checking, Railway Systems, Railway Model Checking, Hyperloop, Modular Transportation

INTRODUCTION

Change is scary. Technological change is no exception to this rule. Fear of developing technology is not new and is not going away despite the breakneck speed with which it is changing society. Around the time of the invention of steam locomotives, people were nervous about things that today might seem silly. Some worried that women's "uteruses would fly out of [their] bodies" on trains or that people might "simply melt at high speeds" [1]. Gradually, these fears subsided and people realized that train travel was not as scary as was originally thought. However, natural adjustments to change like these require not just time, but time without incident.

Catastrophic mistakes are not confined to an accident's direct victims. They have indirect social effects too. People so frequently fixate on improbable horror stories, that there is an entire category of cognitive biases devoted to the subject called the Neglect of Probability biases [5]. The overemphasis on improbable (yet dramatic) events is part of the reason we spend much more money on anti-terrorism efforts than we do on preventing industrial or infrastructural accidents even though the chances of dying in a car crash is much higher.

The lesson to be learned from this analogy is that if we want to use technology to revolutionize our society, we must do it with as few big accidents as possible. Small accidents can go unnoticed, but large ones in new fields can stunt an entire industry (as Three Mile Island did for a while with Nuclear energy). Change requires trust and big mistakes destroy that trust. Nonetheless, mistakes are inevitable. Therefore, as a society we must put great emphasis on checking for mistakes. With more complex and integrated technology, there is a greater likelihood of accidents with more powerful negative effects. Such events give credence to the nay-sayers and luddites who fear such change to begin with. This paper explores the importance and current status of checking safety-critical software systems with tools like model checking. The first section gives a brief summary of two articles that outline the status of model checking today. The second explores the growing importance of model checking with railway systems, the specific tools used today, and how those tools may evolve. Finally, concluding remarks reinforce the need for model checking as society develops.

I. THE STATUS OF MODEL CHECKING: A LITERATURE REVIEW

A. Review: Formal Methods: Practice & Experience

Woodcock et al. discuss the status of model checking in industrial development and academia in "Formal Methods: Practice and Experience." This article uses literature reviews and a survey of projects to outline how model checking is used in different industries and argues for a more well-defined understanding of the cost-effectiveness of model checking in those industries. Woodcock et al. notes that successful projects usually have well-defined initial goals based on the verification of specifications [8]. These are commonly referred to as *shell statements*, and must be weighed against *environmental constraints* which frame the specifications in terms of possible states and transitions [2]. The process of defining these parameters is often quite time-intensive and can represent a significant portion of the cost associated with model-checking projects [8].

Model Checking projects are motivated by a variety of reasons, including regulations, cost-effectiveness, and reputation. With hardware, it is easier to see the importance of verification. Recalls can be quite expensive when compared to software updates. Despite the differences in cost between hardware and software verification, the latter is not stagnant, as costs can be reduced through verification in other ways [6]. Unfortunately, there still remains an overall "chasm between academics who see formal methods as inevitable and practitioners who see formal methods as irrelevant." [8]

The survey of projects performed by Woodcock is illuminating, but is ultimately limited. Only 62 industrial projects are explored, and of those only 56 were the result of direct survey respondents. While a clear picture is offered of these different projects, one is left wondering how representative these projects are of the overall trends in verification technologies. Nonetheless, the article confidently

asserts a growing trend in model checking, (13% in the 90s to 51% after 2000). It also points to anecdotal information about the cost-effectiveness of model-checking where participants speculate on overall reduced costs. However, as the authors readily admit, hard evidence to support these claims is not easy to find. The survey's main objective is to highlight certain tools and their use in specific industries. For example, SPIN's use in checking for logical consistency in the Mars Rover Exploration project, or SLAM's use in certifying that certain sequencing bugs are not present in Windows device drivers is discussed. Mondex Smart Cards, AAMP Microprocessors, Airbus software, Storm Surge Barriers, Security Systems, and Chip Firmware are other examples of industries explored in the article [8].

The authors note that model checkers have not seen "widespread adoption" except as part of the "development of critical systems in certain domains." This probably relates to the challenge engineers face in identifying how best to balance safety concerns with costs, as not all types of software or software components are safety-critical. Furthermore, verification processes require significant human expertise and many tools lack adequate support for widespread commercial adoption. The authors argue that a more clearly defined cost-benefit equation, a repository of already verified projects, and perhaps more user-friendly, supported tools will increase the adoption of model checkers in industry [8].

B. Review: Survey of Existing Tools for Formal Verification

Another article that explores the current state of formal verification tools and their industrial application is "Survey of Existing Tools for Formal Verification" by Armstrong et al. This article helps clarify several important distinctions between types of approaches and tools used by industry experts. The authors make it clear that different tools serve very specialized purposes. For example, whether one's project involves distributed nodes, synchronous interchanges, or certain programming languages can make a significant difference in the kind of tools that are appropriate. SPIN for example is great for asynchronous processes, while SMV and NuSMV are used to model "synchronous digital logic" systems. Alloy is a useful tool for identifying consistency issues, while Simulink Design Verifier can be important for finding issues with data flow [6].

Further distinctions are important to draw when evaluating the potential for state-space explosion. If the goal is to verify localized components with limited state-space possibilities, explicit modeling may be preferable as it more comprehensively evaluates the computations occurring in each state. Implicit modeling on the other hand, evaluates the "properties of a function, rather than its execution," and is based on pre and post conditions. Automated model checkers use "algorithmic shortcuts" to evaluate properties exhaustively, while theorem provers depend more heavily on human involvement guide the process [6].

Some systems benefit more from being designed with verification in mind, while others are usually verified afterwards. Hardware, which has a much longer history of verification than software, often requires simulation and testing via abstract and explicit state methods to minimize the risk of recall [6].

Finally, the cost of verification is another important consideration, as licensing these tools can be also be quite expensive and certain approaches require significant time spent by human experts. Thus, successful checkers will spend considerable time weighing the pros and cons of different approaches before beginning a verification process. Sometimes this requires making decisions about verification before the software is even designed [6].

As both articles are surveys of different project types and tools, it is beyond the scope of this document to outline each in turn. Rather, the next section explores a subject both discuss and agree upon the importance of: railway systems.

II. Verifying Railway Systems

Perhaps one of the most visible manifestations of technological transformation in society has been the revolutions in transportation over the last century and a half. In about 150 years, people went from traveling by horse to traveling in space. Visiting countries on distant parts of the world used to take years, but now can be done in mere hours. Even more impressive is that we are far from finished in revolutionizing our transportation systems. Whether avionics, automotive, space travel, or railway all systems are undergoing development.

In the railway sector, two new technologies hold particularly exciting possibilities: the hyperloop and modular transportation. Currently, the hyperloop is in early design phases, but is slated to be tested in California, England, and other places. Some speculate this type of transport system can travel at speeds as high as 800 miles per hour (that's about 200 mph faster than the average plane) and can be quite efficient. The hyperloop technology involves traveling in pods that move through vacuum tubes [4]. This approach reduces friction significantly allowing for faster speeds and more efficient travel. Furthermore, tubes can reduce some environmental risks. However, because speeds are higher and tubes are vacuums, accidents have the potential to be quite catastrophic.

Modular transport has similar potential to revolutionize transportation. Currently, people who want to travel long distances must use multiple means, (bus/car to airport, then plane, then car etc.). Modular transport argues for transport designs in which people share transport for portions of a trip, then separate into distinct pods which travel on roadways to a person's specific location (door-to-door). This in effect allows for the sharing of energy and momentum for long-distance parts of the trip and reduces congestion overall on transport mediums.

One can speculate that the future of transportation could mean using a public transport system that takes you from your front door to your destination's door in a distant city without any transfers while traveling at speeds faster than average planes offer today. Unfortunately, for such a scenario to become realistic, extremely complex computer systems will need to be designed to ferry people in reliable and safe ways.

Thus, technology that verifies the reliability of software used by such systems is extremely important. Whatever verification systems do eventually develop for this industry, they will most likely be derivative of current railway verification technologies.

Both Woodcock et al. and Armstrong et al. explore railway systems as examples of industries with a history of using verification technologies. Woodcock mentions the use of B in the development of the SACEM system used by the MATRA Transport and RATP transport in Paris. Armstrong et al. emphasizes the Rodin/Event B approach to designing switches for railways systems. Unlike "after the fact" systems which verify software that has already been created, these systems are used in the creation of software systems. This approach requires one to develop a verified abstract initial model, and then to refine that model as software development progresses. In each stage of development, specifications are verified ensuring that the product is completely verified by its final stages [6],[8].

Unfortunately, Armstrong et al. points out that while such systems are excellent at maintaining "safety properties," they fail to live up to "liveness properties" which are typically verified by many "after the fact" tools. Furthermore, this approach suffers from being flat (non-hierarchical) and can be difficult to compose and decompose. Finally, this approach is restricted to just the one closed-source theorem prover and is also limited in its ability to deal with notions of time [6].

Garmhausen et al. in "Verification of a Safety-Critical Railway Interlocking System with Real-Time Constraints" explores an alternative approach using the Verus system. This formal verification tool "combines symbolic model checking and quantitative timing analysis." Verus uses "discrete notions of time" that allow it to assess transitions and states from a temporal dimension and then locate inefficiencies in time as a side-effect. Furthermore, the Verus system is quite time-efficient with large state-space systems (performing a detailed check of a system with 10^{27} states in only minutes). The article explores the use of this system for verifying the ACC Transport system which uses three different computers on a "2 out of 3" methodology to communicate with physical devices on a distributed system [3].

As railway systems develop and incorporate more complex distributed system management technologies, verification will be increasingly important and challenging. It is likely that any system which can handle the complexity of a modular hyperloop transport system will need to deal with a considerable number of vulnerabilities. These vulnerabilities could involve hazards as serious as violent acts of terror on tubes ferrying pods to as simple as errors that result in ferrying people to the wrong location. Advanced systems will most likely need to be developed with verification in mind as the Rodin/Event-B systems discussed by Armstrong and Woodcock do. They might also need to be verified "after the fact" with tools like Verus. Furthermore, systems like the ones used in Hong Kong MTR use machine learning techniques to handle unexpected scenarios as they arise [7]. Such smart systems would anticipate problems more quickly than human observers might be able to, and then could act accordingly with the safety protocols previously verified in the design process and afterwards.

Conclusion

One can imagine the shock experienced by the same person who worried individuals might melt from steam locomotive velocities if they were to view today's complex infrastructure. It then begs questions about our own potential shock when viewing the transportation systems of the not too distant future. How that system manifests will not be obvious nor perfect. Mistakes are inevitable, but learning from them and preventing them is not, which is why we must prioritize such pursuits. With increased integration of model checking technology throughout various industries, progress in speed and quality of safety checks will improve. Without such improvement, unacceptable mistakes will not only cost lives in the near term but may stunt the development of technologies that could save and improve lives in decades to come. It is easy to see such safety checks as cost-prohibitive, but wise to see them as ultimately necessary. For that reason it is important computer scientists and industry experts remain vigilant in the development of these crucial technologies.

References

[1] Burns, Janet. "Early Trains Were Thought to Make Women's Uteruses Fly Out." Editorial. Mental Floss. N.p., n.d. Web. 21 Apr. 2017. <http://mentalfloss.com/article/67806/early-trains-were-thought-make-womens-uteruses-fly-out>.

[2] Cofer, Darren, Michael Whalen, and Steven Miller. "Software Model Checking for Avionics Systems." 2008 IEEE/AIAA 27th Digital Avionics Systems Conference (2008): n. pag. Web.

[3] Garmhausen, Vicky, Sergio Campos, Alessandro Cimatti, Edmund Clarke, and Fausto Giunchiglia. "Verification of a Safety-Critical Railway Interlocking System with Real-Time Constraints." Science of Computer Programming (2000): n. pag. Web. 23 Apr. 2017.

[4] Hughes, David. "What Is the Hyperloop, How Fast Can Elon Musk's Transport System Go and Will It Come to the UK?" The Sun. The Sun, 07 Apr. 2017. Web. 25 Apr. 2017. <https://www.thesun.co.uk/tech/2142155/hyperloop-elon-musk-how-fast-uk-appearance/>.

[5] Kahneman, Daniel. Thinking, Fast and Slow. New York: Farrar, Straus and Giroux, 2015. Print.

[6] Punnoose, Ratish J., Robert C. Armstrong, Matthew H. Wong, and Mayo Jackson. "Survey of Existing Tools for Formal Verification." (2014): n. pag. Web.

[7] Wong, Maggie Hiufu. "The World's Most Envied Metro System." CNN. Cable News Network, 31 Mar. 2015. Web. 24 Apr. 2017. <http://www.cnn.com/2015/03/29/travel/hong-kong-mtr-success-story/>.

[8] Woodcock, Jim, Peter Gorm Larsen, Juan Bicarregui, and John Fitzgerald. "Formal Methods." ACM Computing Surveys 41.4 (2009): 1-36. Web.