Conference Paper Title*

Taten H. Knight
College of Aviation, Science and Technology
Lewis University
Romeoville, USA
knight.taten@gmail.com

***Abstract*—This paper is a literature review of *Comparison of Model Checking Tools*[1] and *Introduction to Embedded Software Verification*[2]. It explores the contributions of each paper to the study of model checking — one of various formal software verification methods. The papers above explore the application of model checking to Embedded Systems and Information Systems (IS). I examine the novel ideas and methods presented, the tools used, the areas of application relevant to the given information and methods, and the current and potential applications of the research to my own professional activities.**

***Index Terms*—component, formatting, style, styling, insert**

I. Introduction

Formal verification is the process of verifying a software system by exploring the state-space of the system and confirming that it meets a set of specifications for all of the states. Model checking is a method of formal verification wherein a model checking software creates a model of the original software that allows it to be explored and verified using a formal logic language (CTL, LTL) [1][2]. Formal software verification methods are more pertinent as the size and complexity of software systems continues to grow and maintaining sufficient test coverage across the software is infeasible. At this point (2022) many major manufacturers such as Dassault-Aviation and Airbus are using formal verification on safety and system critical systems [3], however, at the time of the articles being written formal verification had not been widely implented in industry.

II. Comparison of Model Checking Tools for Information Systems

A. Article Introduction

The focus of this paper was the analysis of the performance of six model checking tools on a single case study. Performance was measured based on ease of specifying the information system (IS) properties and behavior, and on the number of IS instances that can be checked.

B. Areas of Application

This article focuses on the use of model checking for a simple IS. An IS is the software and hardware system that supports a data-intensive application. The IS used for this case study was an extremely small-scale model of a typical IS, a library system.

C. Tools

The authors and researchers focused on the comparison of six tools for model-checking:

1) Spin - One of the first model checkers developed. It introduced the classical approach for on-the-fly Linear Temportal Logic (LTL) model checking. It uses propositional LTL with the *always*, *eventually*, and *until* operators. Spin has no concept of events, and a run consists of a trace with the states visited during execution.
2) NuSMV - The first implementation of Symbolic Model Checking, which uses a symbolic representation of the software to check against the specifications. It is able to deal with Computational Tree Logic (CTL), LTL, and SAT-based bounded model checking. It uses the Promela language, which is very low-level, and can result in it taking longer to write specifications. It can also be either state or event oriented, which gives flexibility.
3) fdr2 - A state model checker for Communicating Concurrent Processes (CSP). It "can check refinement, deadlocks, livelocks, and determinancy of process expressions. It builds and then compresses the state-space transition graph, which makes it an implicit model checker.
4) cadp - Uses the LOTOS NT language to specify models and XTL to specify properties. It does not naturally support state variables which can make model checking difficult.
5) Alloy - Alloy is another symbolic model checker. All properties and specifications are expressed as first-order formulae.
6) ProB - A very flexible model checker that uses the B method. "Properties in ProB can be written in LTL, past LTL or CTL, hence combining the strengths of each language." It allows for "the inclusion of first-order formulae in temporal formulae.

D. New Concepts and Knowledge

I believe that the conclusion of the case study provided the most insight into model checker viability. A good model checker needs to be able to handle both state and events. Pure LTL is useful, but cannot be the sole language used. A pure first-order logic like Alloy can be used but may not be the most intuitive. ProB is the most 'polyvalent' model checker, meaning that it handles the most situations with model checking.

E. Professional Applicability

In my current professional career as a full-stack developer, the above methods are useful for potentially testing backend infrastructure and verifying that data integrity is maintained through backups, updates, deletes, or other forms of manipulations. The authors note that user interaction (including queries) are not covered in this study, so these specific methods may not be applicable for the frontend to backend to frontend flow of the full software stack.

III. Introduction to Embedded Software Verification

A. Article Introduction

Introduction to Embedded Software Verification is a solid introduction to software verification in general. It covers the core concept of software verification and two of its subsets; model checking and theorem proving. It also provides an overview of how one would approach model checking on a microcontroller.

B. Areas of Application

Because this paper provides an introduction to the general concept of software verification, it can provide a precursory look into the concepts before diving into more in-depth research and literature. I would argue that because of this, the paper is widely applicable to the field as an introduction. It also covers the concept of CTL, which is used in a few of the tools mentioned in [1]. I plan on using it as a reference for reviewing easy concepts before moving onto others research.

C. Tools

The paper is a conceptual overview and not a research paper, so there are no technical tools used. That being said, the paper does utilize some notable pedagogical tools, namely repetition and summarization to re-emphasize and solidify key points. It also shallowly explores the broader field of embedded software verification through model checking assembly code. This is a preferred technique for embedded software because once the model checker is developed for a microcontroller or similar system, it can be reused across any software that is installed on that system.

D. New Concepts and Knowledge

Using assembly code model checkers for microcontrollers and similar subsystems is and will continue to prove a valuable and useful concept for any and all large scale custom silicon and controller manufacturers. "...[V]erification effort of nowadays embedded systems projects consumes up to 70 percent of the entire engineering budget" "unpublished" [2]. Anything to reduce average overhead on development and verification of software is going to rightfully result in extreme savings.

E. Professional Applicability

Assembly level verification borders on the realm between computer science and computer engineering in my mind, and my current role doesn't sit on this border. I am, however, focusing on AI/ML in my studies, and the field as a whole is and will likely continue to move into custom silicon for AI/ML workloads. These chips will need to be verified as individual components and as part of the hardware/software stack that they are being included in. As these applications grow, having a combination of verification methods for these chips will be a necessity.

Looking even further to the future, verification of massive software systems may be made even easier with the eventual advent of quantum computing systems.

IV. Conclusion

V. Prepare Your Paper Before Styling

Before you begin to format your paper, first write and save the content as a separate text file. Complete all content and organizational editing before formatting. Please note sections IV-A–IV-E below for more information on proofreading, spelling and grammar.

Keep your text and graphic files separate until after the text has been formatted and styled. Do not number text heads—LaTeX will do that for you.

A. Abbreviations and Acronyms

Define abbreviations and acronyms the first time they are used in the text, even after they have been defined in the abstract. Abbreviations such as IEEE, SI, MKS, CGS, ac, dc, and rms do not have to be defined. Do not use abbreviations in the title or heads unless they are unavoidable.

B. Units

- Use either SI (MKS) or CGS as primary units. (SI units are encouraged.) English units may be used as secondary units (in parentheses). An exception would be the use of English units as identifiers in trade, such as "3.5-inch disk drive".
- Avoid combining SI and CGS units, such as current in amperes and magnetic field in oersteds. This often leads to confusion because equations do not balance dimensionally. If you must use mixed units, clearly state the units for each quantity that you use in an equation.
- Do not mix complete spellings and abbreviations of units: "Wb/m^2" or "webers per square meter", not "webers/m^2". Spell out units when they appear in text: ". . . a few henries", not ". . . a few H".
- Use a zero before decimal points: "0.25", not ".25". Use "cm^3", not "cc".)

C. Equations

Number equations consecutively. To make your equations more compact, you may use the solidus (/), the exp function, or appropriate exponents. Italicize Roman symbols for quantities and variables, but not Greek symbols. Use a

long dash rather than a hyphen for a minus sign. Punctuate equations with commas or periods when they are part of a sentence, as in:

$$a + b = \gamma \tag{1}$$

Be sure that the symbols in your equation have been defined before or immediately following the equation. Use "(1)", not "Eq. (1)" or "equation (1)", except at the beginning of a sentence: "Equation (1) is . . ."

D. LaTeX-Specific Advice

Please use "soft" (e.g., `\eqref{Eq}`) cross references instead of "hard" references (e.g., `(1)`). That will make it possible to combine sections, add equations, or change the order of figures or citations without having to go through the file line by line.

Please don't use the `{eqnarray}` equation environment. Use `{align}` or `{IEEEeqnarray}` instead. The `{eqnarray}` environment leaves unsightly spaces around relation symbols.

Please note that the `{subequations}` environment in LaTeX will increment the main equation counter even when there are no equation numbers displayed. If you forget that, you might write an article in which the equation numbers skip from (17) to (20), causing the copy editors to wonder if you've discovered a new method of counting.

BibTeX does not work by magic. It doesn't get the bibliographic data from thin air but from .bib files. If you use BibTeX to produce a bibliography you must send the .bib files.

LaTeX can't read your mind. If you assign the same label to a subsubsection and a table, you might find that Table I has been cross referenced as Table IV-B3.

LaTeX does not have precognitive abilities. If you put a `\label` command before the command that updates the counter it's supposed to be using, the label will pick up the last counter to be cross referenced instead. In particular, a `\label` command should not go before the caption of a figure or a table.

Do not use `\nonumber` inside the `{array}` environment. It will not stop equation numbers inside `{array}` (there won't be any anyway) and it might stop a wanted equation number in the surrounding equation.

E. Some Common Mistakes

- The word "data" is plural, not singular.
- The subscript for the permeability of vacuum μ_0, and other common scientific constants, is zero with subscript formatting, not a lowercase letter "o".
- In American English, commas, semicolons, periods, question and exclamation marks are located within quotation marks only when a complete thought or name is cited, such as a title or full quotation. When quotation marks are used, instead of a bold or italic typeface, to highlight a word or phrase, punctuation should appear outside of the quotation marks. A parenthetical phrase or statement at the end of a sentence is punctuated outside of the closing parenthesis (like this). (A parenthetical sentence is punctuated within the parentheses.)
- A graph within a graph is an "inset", not an "insert". The word alternatively is preferred to the word "alternately" (unless you really mean something that alternates).
- Do not use the word "essentially" to mean "approximately" or "effectively".
- In your paper title, if the words "that uses" can accurately replace the word "using", capitalize the "u"; if not, keep using lower-cased.
- Be aware of the different meanings of the homophones "affect" and "effect", "complement" and "compliment", "discreet" and "discrete", "principal" and "principle".
- Do not confuse "imply" and "infer".
- The prefix "non" is not a word; it should be joined to the word it modifies, usually without a hyphen.
- There is no period after the "et" in the Latin abbreviation "et al.".
- The abbreviation "i.e." means "that is", and the abbreviation "e.g." means "for example".

An excellent style manual for science writers is [**?**].

F. Authors and Affiliations

The class file is designed for, but not limited to, six authors. A minimum of one author is required for all conference articles. Author names should be listed starting from left to right and then moving down to the next line. This is the author sequence that will be used in future citations and by indexing services. Names should not be listed in columns nor group by affiliation. Please keep your affiliations as succinct as possible (for example, do not differentiate among departments of the same organization).

G. Identify the Headings

Headings, or heads, are organizational devices that guide the reader through your paper. There are two types: component heads and text heads.

Component heads identify the different components of your paper and are not topically subordinate to each other. Examples include Acknowledgments and References and, for these, the correct style to use is "Heading 5". Use "figure caption" for your Figure captions, and "table head" for your table title. Run-in heads, such as "Abstract", will require you to apply a style (in this case, italic) in addition to the style provided by the drop down menu to differentiate the head from the text.

Text heads organize the topics on a relational, hierarchical basis. For example, the paper title is the primary text head because all subsequent material relates and elaborates on this one topic. If there are two or more sub-topics, the next level head (uppercase Roman numerals) should be used and, conversely, if there are not at least two sub-topics, then no subheads should be introduced.

H. Figures and Tables

a) Positioning Figures and Tables: Place figures and tables at the top and bottom of columns. Avoid placing them

in the middle of columns. Large figures and tables may span across both columns. Figure captions should be below the figures; table heads should appear above the tables. Insert figures and tables after they are cited in the text. Use the abbreviation "Fig. 1", even at the beginning of a sentence.

TABLE I
TABLE TYPE STYLES

Table Head	Table Column Head		
	Table column subhead	*Subhead*	*Subhead*
copy	More table copy[a]		

[a]Sample of a Table footnote.

Fig. 1. Example of a figure caption.

Figure Labels: Use 8 point Times New Roman for Figure labels. Use words rather than symbols or abbreviations when writing Figure axis labels to avoid confusing the reader. As an example, write the quantity "Magnetization", or "Magnetization, M", not just "M". If including units in the label, present them within parentheses. Do not label axes only with units. In the example, write "Magnetization (A/m)" or "Magnetization {A[m(1)]}", not just "A/m". Do not label axes with a ratio of quantities and units. For example, write "Temperature (K)", not "Temperature/K".

ACKNOWLEDGMENT

The preferred spelling of the word "acknowledgment" in America is without an "e" after the "g". Avoid the stilted expression "one of us (R. B. G.) thanks ...". Instead, try "R. B. G. thanks...". Put sponsor acknowledgments in the unnumbered footnote on the first page.

REFERENCES

Please number citations consecutively within brackets [1]. The sentence punctuation follows the bracket [2]. Refer simply to the reference number, as in [3]—do not use "Ref. [3]" or "reference [3]" except at the beginning of a sentence: "Reference [3] was the first ..."

Number footnotes separately in superscripts. Place the actual footnote at the bottom of the column in which it was cited. Do not put footnotes in the abstract or reference list. Use letters for table footnotes.

Unless there are six authors or more give all authors' names; do not use "et al.". Papers that have not been published, even if they have been submitted for publication, should be cited as "unpublished" [**?**]. Papers that have been accepted for publication should be cited as "in press" [**?**]. Capitalize only the first word in a paper title, except for proper nouns and element symbols.

For papers published in translation journals, please give the English citation first, followed by the original foreign-language citation [**?**].

REFERENCES

[1] M. Frappier, B. Fraikin, R. Chossart, R. Chane-Yack-Fa, and M. Ouenzar, "Comparison of Model Checking Tools for Information Systems," International Conference on Formal Engineering Methods. Springer, Berlin, Heidelberg, 2010.
[2] T. Reinbacher, "Introduction to Embedded Software Verification," unpublished
[3] Y. Moy, E. Ledinot, H. Delseny, V. Wiels and B. Monate, "Testing or Formal Verification: DO-178C Alternatives and Industrial Experience" in IEEE Software, vol. 30, no. 03, pp. 50-57, 2013.