Model Checking Clinical Decision Support Systems Using SMT

Mohammad Hekmatnejad¹, Andrew M. Simms²,³, and Georgios Fainekos¹

¹Arizona State University, Tempe AZ 85281, USA,  
{mhekmatn,fainekos}@asu.edu
²Cognitive Medical Systems, San Diego CA 92121, USA,  
³University of Washington, Seattle WA 98195, USA,  
{asimms}@cognitivemedicine.com

Abstract. Individual clinical Knowledge Artifacts (KA) are designed to be used in Clinical Decision Support (CDS) systems at the point of care for delivery of safe, evidence-based care in modern healthcare systems. For formal authoring of a KA, syntax verification and validation is guaranteed by the grammar. However, there are no methods for semantic verification. Any semantic fallacy may lead to rejection of the outcomes by care providers. As a first step toward solving this problem, we present a framework for translating the logical segments of KAs into Satisfiability Modulo Theory (SMT) models. We present the effectiveness and efficiency of our work by automatically translating the logic fragment of publicly available KAs and verifying them using Z3 SMT solver.

Keywords: symbolic model checking, clinical knowledge artifacts, clinical decision support systems

1 Introduction

There are substantial ongoing efforts within the healthcare domain looking to improve healthcare quality by helping patients and providers make better decisions whilst avoiding poor ones. Clinical Decision Support (CDS) systems that help patients track, monitor and optimize the care provided to them, or tools that assist providers in analyzing diagnostic data and developing therapeutic plans are among the goals of current research [5]. Evaluating the interaction between such systems, their users, and actual/intended outcomes is an area of particular interest not only to system developers, but also to regulatory organizations such as the FDA, which is responsible for ensuring patient safety. Whether embedded as control logic within a medical device, or deployed as a service component in server, individual CDS knowledge artifacts (models) are often developed in order to optimize health outcomes and efficiency within the context of a specific care process.

Model checking methods, which were developed for software and hardware verification, have the potential to address the complexity of the aforementioned
problem. Model checking is a method by which a modeled system can be validated for compliance against a set of pre-defined specifications. Domain knowledge, such as knowledge artifacts incorporated into a software system, is also an artifact that can be model checked. Indeed, any software architected around a corpus of rapidly evolving domain knowledge creates numerous possibilities for model inconsistencies to arise. When such artifacts are bundled into more complex systems, or reused in settings different from those assumed by the original scenario, the resulting artifact must be carefully evaluated to ensure that it is still logically and clinically suitable for the new context of care. We propose to utilize Satisfiability Modulo Theory (SMT) based model checking as an effective approach for evaluating KAs.

Related work: Among all different research lines for the use of Artificial Intelligence (AI) in medicine, there are notable works dedicated to verification of medical device systems [10,9], model checking Clinical Guidelines (CG) and Clinical Pathways [13]. Formal verification of CGs is not a new research area, and some works such as in [11] modeled the content of CG and verified their properties using different techniques. While most of the previous works had to model human readable CGs, we work with some machine readable, standardized and formally defined evidenced base knowledge artifacts. Here, we are interested in verifying the satisfiability of formalized logical statements in KAs.

There exist successful prior arts on the verification of other classes of rule-based systems using SMT verification [14], and there already exist methods for computing the minimal set of inconsistent SMT formulas [2], which justify our approach. SMT based model checking has been used for a long history of success in both academia and industry [4], and we use it in validating KAs.

2 Problem

During the knowledge authoring phase, knowledge engineers and designers translate and bundle clinical knowledge components into KAs. The resulting artifact must be carefully evaluated to ensure that it is logically consistent and clinically suitable for the context of care. From a knowledge management and governance perspective this task becomes exponentially onerous as the number of combinatorial possibilities increases. Knowledge artifacts can be checked for inconsistencies and conflicts during the authoring phase by using established model verification techniques and tools. For example, a KA about heart failure, may have conditions attached to specific actions so that if they become satisfiable, then their corresponding actions, such as prescribing specific medication orders or procedures can be executed. These conditions usually are complicated logical expressions that are translated from English narrative to a formal language. There is always a chance that the actual narrative logic or its translated formalization (ELM expression logics) is not sound.

We propose a framework that exploits SMT solvers [3] to analyze satisfiability of some specifications \( \varphi \) (given by domain experts) and expression logics \( \mathcal{L} \) of CDS KAs by translating them into SMT formulas \( \mathcal{M} \). If no instance model was
attainable, then it means the expression logic is not satisfiable, and therefore unsatisfiable expressions are detectable. Our proposed solution effectively and efficiently analyzes the embedded logics of CDS KAs.

**Problem Formal Definition** Given a knowledge artifact \( \mathcal{K} \), we are interested in translating expression logics \( \mathcal{L} \) embedded in \( \mathcal{K} \) into an equivalent formula \( \mathcal{M} \) to check for existence of an instance model \( \mathcal{S} \), such that \( \mathcal{S} \) satisfies \( \mathcal{M} \) with respect to a given specification formula \( \varphi \).

### 3 Preliminaries

**Satisfiability Modulo Theories (SMT):** The SMT problem is checking if a given closed logical formula \( \varphi \) is satisfiable with respect to some background theory \( \mathcal{T} \) which restricts the range of used symbols in \( \varphi \). In other words, the SMT problem for \( \varphi \) and \( \mathcal{T} \) is about existence of models of \( \mathcal{T} \) that satisfy the formula \( \varphi \). An SMT solver is a software that implements a procedure for satisfiability modulo some given theory. SMT solvers come with different underlying logics, background theories, input formulas and interfaces. In this paper, we use the high-performance SMT Solver Z3 that supports all the theories that we need for modeling such as empty theory, linear arithmetic, nonlinear arithmetic, bit-vectors, arrays, data-types, quantifiers and strings.

**CDS Knowledge Artifacts:** We will consider CDS KAs based on the HL7 Knowledge Artifact Specification, which is a XML based container for representing clinical knowledge. The Clinical Quality Language (CQL) is to represent procedural logic and functions within a KA. CQL is written in an XML format called Expression Logical Model (ELM) when written inside a KA. KAs are categorized based on their application into three types: Event Condition Action (ECA) Rules, Documentation Templates, and Order Sets. Beside their different types, all KAs use the same components as their building blocks. In a KA, the only executable components are actions, and in order to see if they can be executed during run-time, we should check if their control conditions evaluate to \( \text{true} \). In the next section, we assume that the goal is to check if all the underlying Expression Logics represented in ELM are satisfiable despite to coming data in the executing time.

### 4 Example of Translating and Verifying A Sample KA

In this section, we only show some ELM operators and their equivalent SMT in examples. We are going to use an OS for “heart failure admission to medical/surgical unit” as a running example mainly because it has a simple expression logic with the least number of medical terms. This OS has only one condition attached to an action group that has one simple action. Here, we are not focusing on finding contradictions in actions or the contradictions that may occur as a result of executing them as those are out of scope of this paper. In List 1.1 the Expression Logic of the OS is depicted. It can be noticed that there is a logical \( \text{AND} \) between two expressions, one is an equality expression and the other
is a sequence of logical NOT and Exist operators. This condition in English means “If the age of patient (evaluated in years) is greater than or equal to 18, and the patient has no history of adverse reaction to ACE inhibitors, then...”.

We state the equivalent SMT code of the logic in the List 1.2. There is an assertion in line 6 that is semantically equivalent to the aforementioned logic in List 1.1 as all the ELM operators have equivalent operators in SMT language. We defined the Exist operator as a function in line 4, and for the variables, AdverseReactionToACEInhibitors and PatientAgeInYears are constants of types List of AdverseEvent and Integer, respectively. Also, AdverseEvent is not a primitive data type nor a complex one; therefore, it is declared as a sort in SMT. We do not go into details of how these types are decided in the translator, we just mention that the translator extracted them using other sections of KA such as external data and expressions.

For the sake of simplicity, one can check that the logic in List 1.1 can be represented as \((\text{And} (\geq \text{PatientAgeInYears} 18) (\text{Not} (\text{Exists} \text{AdverseReactionToACEInhibitors})))\). This is almost the same code as is stated in line 6 of List 1.2, just with an extra SMT assertion with template “\((\text{assert (= true (ELM expression logic)))}\)” to check if the logic statement is satisfiable.

Note, here we chose a simple use-case to describe the problem and solution, but KAs can have large and complicated embedded logics each with tens of operators and symbols, such as logics that describe identification of Sepsis and Systemic Inflammatory Response Syndrome (SIRS) ICU [8][12].

We discovered one KA with no satisfiable model from HL7 CDS KA release 1.4.1. This is an ECA Rule with a condition in which there is an inequality statement around patient’s age informally stated as \((\geq 18 \text{ PatientAge}) \text{ And } (<=...
which is logically a wrong statement. In the SMT translation of the code, SMT solver reported the rule (tagged by name “assertion-1”) as part of the unsatisfiable core of the code. We added set-option:produce-unsat-cores true as a Z3 configuration command to force the solver to detect the unsatisfiable assertions, and used (get unsat core) for reporting them.

5 Experimental Results

The concept of CDS KAs is new even to the CDS community. Therefore, there are not many KAs available for testing. In the Table 1, the execution time for seven of the publicly available KAs is presented.

| KA Name | Expr | Oper | Prep | Tran | Solv |
|---------|------|------|------|------|------|
| OS-01   | 6    | 3406 | 26   | 125  |
| ECA-01  | 15   | 3885 | 79   | 148  |
| ECA-02  | 19   | 4128 | 92   | 122  |
| ECA-03  | 9    | 3946 | 64   | 149  |
| ECA-04  | 6    | 3668 | 47   | 117  |
| DT-01   | 1    | 3228 | 28   | 180  |
| DT-02   | 3    | 4135 | 30   | 105  |

Table 1: Running time in milliseconds. All the cases except the ECA-03 were satisfiable.

In Table 2, some of the ELM operators that our framework supports for one-to-one translation to SMT are listed by their categories and support coverage. Note that in Table 1, the higher execution times in the “Preparing” column are efficient for our use-cases because of small size of the models (only conditions). Note that if we translate and incorporate the other elements of a KA, such as actions and behaviors, then based on our experiments, the execution time is potentially going to rise, but we expect it to remain feasible for quantitative model solvers.

In Table 2, some of the ELM operators that our framework supports for one-to-one translation to SMT are listed by their categories and support coverage. Note that in Table 1, the higher execution times in the “Preparing” column are because of loading all the schema files, verifying artifacts against them, and then unmarshalling them into Java instance objects. Our translation tool is currently in a beta version and publicly available at: [https://cpslab.assembla.com/spaces/cqlverifier](https://cpslab.assembla.com/spaces/cqlverifier) under GNU GPLv3 license.

6 Results and Future Work

The preliminary results reported here revealed that even KAs with simple logics may have fallacies in them, which need to be fixed in the knowledge authoring phase. For example, “ECA-03” in Table 1 has two inequality statements about patient’s age which are contradictory. Second, we found out that the current HL7 KNART specification does not support range constraints on variables explicitly. For example, while the age of a patient is considered as an Integer variable, there is no assumption regarding the valid age of a live person.
We intend to apply our tools to a larger and more complex set of knowledge artifacts currently under development that cover multiple clinical specialties, and a broad range of complexity. Further, we plan to extend the translation capability to the complete definition of ELM expressions described in the standard.

Acknowledgements The authors thank Cognitive Medical Systems, Inc. for their support and access to their database of KNART artifacts. This work was supported in part by the NSF I/UCRC Center for Embedded Systems and from NSF grant #1361926.

References

1. Barrett, C., Stump, A., Tinelli, C., et al.: The smt-lib standard: Version 2.0. In: Proceedings of the 8th International Workshop on Satisfiability Modulo Theories (Edinburgh, England). vol. 13, p. 14 (2010)
2. Cimatti, A., Griggio, A., Sebastiani, R.: Computing small unsatisfiable cores in satisfiability modulo theories. Journal of Artificial Intelligence Research 40, 701–728 (2011)
3. De Moura, L., Bjørner, N.: Z3: An efficient smt solver. Tools and Algorithms for the Construction and Analysis of Systems pp. 337–340 (2008)
4. De Moura, L., Bjørner, N.: Satisfiability modulo theories: introduction and applications. Communications of the ACM 54(9), 69–77 (2011)
5. Greenes, R.A.: Clinical decision support: the road ahead. Academic Press (2011)
6. Health Level Seven International: HL7 Cross-Paradigm Specification: Clinical Quality Language, Release 1.2
7. Health Level Seven International: HL7 Standard: Clinical Decision Support Knowledge Artifact Specification, Release 1.3, dstu update 3 edn. (Jul 2015), available at: http://www.hl7.org/implement/standards/
8. Kaukonen, K.M., Bailey, M., Pilcher, D., Cooper, D.J., Bellomo, R.: Systemic inflammatory response syndrome criteria in defining severe sepsis. New England Journal of Medicine 372(17), 1629–1638 (2015)
9. Lee, I., Pappas, G.J., Cleaveland, R., Hatcliff, J., Krogh, B.H., Lee, P., Rubin, H., Sha, L.: High-confidence medical device software and systems. Computer 39(4), 33–38 (2006)
10. Murugesan, A., Whalen, M.W., Rayadurgam, S., Heimdahl, M.P.: Compositional verification of a medical device system. In: ACM SIGAda Ada Letters. vol. 33, pp. 51–64. ACM (2013)
11. Pérez, B., Porres, I.: Authoring and verification of clinical guidelines: A model driven approach. Journal of biomedical informatics 43(4), 520–536 (2010)
12. Pittet, D., Rangel-Frausto, S., Li, N., Tarara, D., Costigan, M., Rempe, L., Jebson, P., Wenzel, R.: Systemic inflammatory response syndrome, sepsis, severe sepsis and septic shock: incidence, morbidities and outcomes in surgical icu patients. Intensive care medicine 21(4), 302–309 (1995)
13. Scivicco, G., Juarez, J.M., Campos, M.: Quality checking of medical guidelines using interval temporal logics: A case-study. In: Int Work-Conference on the Interplay Between Natural and Artificial Computation. pp. 158–167. Springer (2009)
14. Vanucchici, C., Diamanti, M., Mazzante, G., Cacciagrano, D., Culmone, R., Gorogiannis, N., Mostarda, L., Rainon, F.: Symbolic verification of event-condition-action rules in intelligent environments. Journal of Reliable Intelligent Environments pp. 1–14 (2017)