第 3 章　Logics for verifying students' learning processes: A survey

Norihiro Kamide[1]

Abstract

In this report, a survey of two papers on logics for verifying students' learning processes in learning support systems is given. These papers were written by the author.

1. Introduction

In this report, a survey of the papers [Kamide 2013a] and [Kamide 2013b], which were written by the author, is given. A motivation of these papers was to formalize students' learning processes within an appropriate logic. Formalizing students' learning process in an appropriate logic is required for implementing verification algorithms in some learning support systems such as *intelligent tutoring* systems [Freedman 2000] [Nwana 1990] and *e-learning* systems.

A model of students in such a system should be inconsistency-tolerant since student's understanding is uncertain and vague in general. Moreover, detailed information on students should be well-structured with hierarchical information. In order to represent such a student model, we need a paraconsistent negation connective, which can appropriately represent inconsistency-tolerant reasoning, and some sequence modal operators, which can suitably represent hierarchical information.

The papers [Kamide 2013a] and [Kamide 2013b] proposed such an appropriate logic. The paper [Kamide 2013a] proposed an extension of the standard *linear-time temporal logic* (LTL) [Pnueli 1977], and the paper [Kamide 2013b] proposed an extension of the standard *computation-tree logic* (CTL) [Emerson and Halpern 1986][Emerson and Sistla 1984].

2. The proposed logic in [Kamide 2013a]

In the paper [Kamide 2013a], a new extended linear-time temporal logic (LTL), called *sequential paraconsistent LTL* (SPLTL), was introduced as a Kripke semantics

[1] Associate Professor, Faculty of IT and Business, Cyber University

with a paraconsistent negation connective and some sequence modal operators. The logic SPLTL can appropriately represent both, inconsistency-tolerant reasoning by the paraconsistent negation connective, and hierarchical information by the sequence modal operators. Some illustrative examples for verifying Students' learning processes can be obtained using SPLTL. Some theorems for embedding SPLTL into a paraconsistent version PLTL of LTL and into LTL were proved. By using these embedding theorems, SPLTL was shown to be decidable.

Theorem (Embeddability):

SPLTL is embeddable into LTL, namely, the following holds: Let f be a mapping from SPLTL into LTL. Then, for any formula α, α is valid in SPLTL iff $f(\alpha)$ is valid in LTL.

Theorem (Decidability):

The validity and satisfiability problems for SPLTL are decidable.

From the point of view of logic, SPLTL is a combination of LTL and *Nelson's paraconsistent four-valued logic with strong negation*, N4 [Nelson 1949]. LTL is known to be one of the most useful temporal logics for verifying and specifying concurrent systems. On the other hand, N4 is known to be one of the most important base logics for inconsistency-tolerant reasoning. Combining the logics LTL and N4 was studied in [Kamide and Wansing 2012], and such a combined logic was called *paraconsistent LTL* (PLTL). Roughly speaking, SPLTL is obtained from PLTL by adding some sequence modal operators.

Combining LTL with some sequence modal operators was studied in [Kaneiwa and Kamide 2010], and such a combined logic was called *sequence-indexed LTL* (SLTL). SPLTL is regarded as a modified paraconsistent extension of SLTL, and hence SPLTL is a modified extension of both PLTL and SLTL. In the following, we explain an important property of paraconsistent negation and a plausible interpretation of sequence modal operators.

One reason why the paraconsistent negation connective ~ in SPLTL is considered is that it may be added in such a way that the extended logics satisfy the property of *paraconsistency*. A semantic consequence relation $\models$ is called paraconsistent with respect to a negation connective ~ if there are formulas α, β such that $\{\alpha, \sim\alpha\} \not\models \beta$. In the case of LTL, this means that there is a model M and a position i of a sequence $\sigma = t_0, t_1, t_2, \ldots$ of time-points in M with $(M, i) \not\models (\alpha \wedge \sim\alpha) \rightarrow \beta$.

It is known that logical systems with paraconsistency can deal with inconsistency-tolerant and uncertainty reasoning more appropriately than systems which are non-paraconsistent. For example, we do not desire that $(s(x) \wedge \sim s(x)) \rightarrow d(x)$

is satisfied for any symptom s and disease d where ~s(x) means ``person x does not have symptom s'' and d(x) means ``person x suffers from disease d'', because there may be situations that support the truth of both s(a) and ~s(a) for some individual a but do not support the truth of d(a). For more information on paraconsistency, see e.g., [Kamide and Wansing 2012][Kamide 2013c].

A sequence modal operator [b] in SPLTL represents a sequence b of symbols. The notion of sequences is useful to represent the notions of ``information,'' ``trees,'' ``orders'' and ``ontologies.'' Thus, ``hierarchical information'' can be represented by sequences. This is plausible because a sequence structure gives a *monoid* (M, ;, ∅) with *informational interpretation* [Wansing 1993]:

1. M is a set of pieces of (ordered or prioritized) information (i.e., a set of sequences),
2. ; is a binary operator (on M) that combines two pieces of information (i.e., a concatenation operator on sequences),
3. ∅ is the empty piece of information (i.e., the empty sequence).

A formula of the form $[b_1 ; b_2 ; \dots ; b_n]\,\alpha$ in SPLTL intuitively means that ``α is true based on a sequence $b_1 ; b_2 ; \dots ; b_n$ of (ordered or prioritized) information pieces.'' Further, a formula of the form $[\emptyset]\alpha$ in SPLTL, which coincides with α, intuitively means that ``α is true without any information (i.e., it is an eternal truth in the sense of classical logic).''

3. The proposed logic in [Kamide 2013b]

In the paper [Kamide 2013b], a new extended computation tree logic (CTL), called *sequential paraconsistent computation tree logic* (SPCTL), was introduced as a Kripke semantics with a paraconsistent negation connective and some sequence modal operators. Some new illustrative examples for students' learning processes were presented using SPCTL. The validity, satisfiability and model checking problems of SPCTL were shown to be EXPTIME-complete, deterministic EXPTIME-complete and deterministic PTIME-complete, respectively. These complexity results were proved using some theorems for embedding SPCTL into a *paraconsistent CTL* (PCTL) and into CTL.

Theorem (Embeddability):
SPCTL is embeddable into CTL, namely, the following holds: Let g be a mapping from SPCTL into CTL. Then, for any formula α, α is valid in SPCTL iff $g(\alpha)$ is valid in CTL.

Theorem (Complexity):
The validity, satisfiability and model-checking problems for SPCTL are

EXPTIME-complete, deterministic EXPTIME-complete and deterministic PTIME-complete, respectively.

These embeddability and complexity results for SPCTL allow us to use the existing CTL-based algorithms to test the satisfiability. Thus, it was shown in the paper [Kamide 2013b] that SPCTL can be used as an executable logic to model and verify inconsistency-tolerant temporal reasoning with hierarchical information.

Compared with the logic SPLTL in [Kamide 2013a], the logic SPCTL in [Kamide 2013b] has an efficient model checking algorithm and is executable by using such an algorithm. The main difference between SPCTL and SPLTL is the base logic: SPCTL is based on CTL, and SPLTL is based on LTL. As well-known, CTL has an efficient model checking algorithm in deterministic PTIME-complete, but a simple specification cannot be given by CTL. On the other hand, LTL can give some simple specifications, but has no efficient model checking algorithm. SPCTL and SPLTL inherited these properties for specification and model-checking for CTL and LTL.

In the rest of this report, we give some illustrative examples based on SPCTL. As mentioned in Section 1, a model of students should be inconsistency-tolerant since student's understanding is uncertain and vague. SPCTL can be used to express the negation of uncertain concepts such as *understand* (or *understanding*). For instance, if we cannot determine whether someone understands, the uncertain concept *understand* can be represented by asserting the inconsistent formula: $understand \wedge \sim understand$. This is well formalized because the formula: $(understand \wedge \sim understand) \rightarrow \perp$ is not valid in paraconsistent logic. On the other hand, we can decide whether someone is learning: The decision is represented by $\sim learning$, where $(learning \wedge \sim learning) \rightarrow \perp$ is valid in classical logic. It is remarked that the following negative expressions can be differently interpreted: $\neg understand$ (does not understand) and $\sim understand$ (does not deeply understand). The first statement indicates that a person does not understand that is inconsistent with his or her understanding. The second statement means that we can say that a person does not deeply understand, but he or she may be in shallowly understanding. We thus allow the situation: $understand \wedge \sim understand$.

In ontology representation, a concept hierarchy is constructed by *ISA-relations* between concepts, i.e., a concept is a subconcept of another concept. In this study, the author used sequence modal operators to represent ISA-relations between concepts. Let $c_1, c_2, \ldots, c_n$ be concept symbols. Then, we write a sequence of concept names by $[c_1; c_2; \ldots; c_n]$. Each order (c_i, c_j) $(1 \leq i < j \leq n)$ of concepts in the sequence modal operator $[c_1; c_2; \ldots; c_n]$ can be used to represent the ISA-relation between c_i and c_j. For example, we declare the following order of two concepts as an ISA-relation between ``human'' and ``student:'' $[student ; human]$. This sequence expresses that the concept ``student'' is a subconcept of the concept ``human.'' The sequence modal operators in SPCTL were applied to hierarchical structures where each hierarchical structure is a specific model of concepts in a hierarchy.

4. Concluding remarks

In this report, the papers [Kamide 2013a] and [Kamide 2013b] on logics for verifying students' learning processes in learning support systems such as intelligent tutoring and e-learning systems were surveyed. In these papers, the logic SPCTL and SPLTL were introduced and studied. It was then explained in this report that SPCTL and SPLTL are appropriate for describing a student model in intelligent tutoring systems.

Finally in this report, we give a remark on intelligent tutoring systems [Freedman 2000] [Nwana 1990] and a student model assumed in this study. Intelligent tutoring systems are computer systems that provide immediate and customized instruction or feedback to students. Intelligent tutoring systems consist of the following four basic components based on general consensus amongst researchers: (1) domain model, (2) student model, (3) tutoring model and (4) user interface model. The discussion in this report was paid special attention to the student model. The student model focuses to students' cognitive and affective states and their evolution as the learning process advances. A student needs step-by-step through their problem solving process. Besides, students' knowledge or understanding is uncertain and vague. Thus, a model of students should be inconsistency-tolerant, and an appropriate logic such as SPCTL and SPLTL is required.

Acknowledgements: This research was partially supported by Cyber University, Grant-in-Aid for e-Learning Research. The papers [Kamide 2013a] and [Kamide 2013b], which were surveyed in this report, were main contributions of this research.

References

[Emerson and Halpern 1986] E.A. Emerson and J.Y. Halpern, "Sometimes" and "not never" revisited: On branching versus linear time temporal logic, *Journal of the ACM* 33 (1), pp. 151–178, 1986.

[Emerson and Sistla 1984] E.A. Emerson and P. Sistla, Deciding full branching time logic, *Information and Control* 61, pp. 175–201, 1984.

[Freedman 2000] R. Freedman, What is an intelligent tutoring system?, *Intelligence* 11 (3), pp. 15-16, 2000.

[Kamide 2013a] N. Kamide, An extended LTL for inconsistency-tolerant reasoning with hierarchical information: Verifying students' learning processes, *International Journal of e-Education, e-Business, e-Management and e-Learning* 3 (3), pp. 234-238, 2013.

[Kamide 2013b] N. Kamide, Modeling and verifying inconsistency-tolerant temporal reasoning with hierarchical information: Dealing with students' learning processes, Proceedings of the 2013 IEEE International Conference on Systems, Man, and Cybernetics (IEEE SMC 2013), pp. 1859-1864, 2013.

[Kamide 2013c] N. Kamide, Formalizing inconsistency-tolerant relevant human reasoning: A decidable paraconsistent relevant logic with constructible falsity, Proceedings of the 2013 IEEE International Conference on Systems, Man, and Cybernetics (IEEE SMC 2013), pp. 1865-1870, 2013.

[Kamide and Wansing 2012] N. Kamide and H. Wansing, Proof theory of Nelson's paraconsistent logic: A uniform perspective, *Theoretical Computer Science* 415, pp. 1-38, 2012.

[Kaneiwa and Kamide 2010] K. Kaneiwa and N. Kamide, Sequence-indexed linear-time temporal logic: Proof system and application, *Applied Artificial Intelligence* 24 (10), pp. 896-913, 2010.

[Nelson 1949] D. Nelson, Constructible falsity, *Journal of Symbolic Logic* 14, pp. 16-26, 1949.

[Nwana 1990] H.S. Nwana, Intelligent tutoring systems: An overview, *Artificial Intelligence Review* 4, pp. 251-277, 1990.

[Pnueli 1977] A. Pnueli, The temporal logic of programs, Proceedings of the 18th IEEE Symposium on Foundations of Computer Science, pp. 46-57, 1977.

[Wansing 1993] H. Wansing, The logic of information structures, *Lecture Notes in Artificial Intelligence* 681, pp. 1-163, 1993.