https://doi.org/10.20965/jaciii.2016.p0813

Paper:

Method for Combining Paraconsistency and Probability in Temporal Reasoning

Norihiro Kamide* **and Daiki Koizumi****

*Faculty of Science and Engineering, Department of Information and Electronic Engineering, Teikyo University
1-1 Toyosatodai, Utsunomiya, Tochigi 320-8551, Japan
E-mail: drnkamide08@kpd.biglobe.ne.jp
**Faculty of Commerce, Department of Information and Management Science, Otaru University of Commerce
3-5-21 Midori, Otaru, Hokkaido 047-8501, Japan
E-mail: dkoizumi@m.ieice.org
[Received June 7, 2016; accepted July 25, 2016]

Computation tree logic (CTL) is known to be one of the most useful temporal logics for verifying concurrent systems by model checking technologies. However, CTL is not sufficient for handling inconsistency-tolerant and probabilistic accounts of concurrent systems. In this paper, a paraconsistent (or inconsistency-tolerant) probabilistic computation tree logic (PpCTL) is derived from an existing probabilistic computation tree logic (pCTL) by adding a paraconsistent negation connective. A theorem for embedding PpCTL into pCTL is proven, thereby indicating that we can reuse existing pCTL-based model checking algorithms. A relative decidability theorem for PpCTL, wherein the decidability of pCTL implies that of PpCTL, is proven using this embedding theorem. Some illustrative examples involving the use of PpCTL are also presented.

Keywords: computation tree logic, paraconsistent reasoning, probabilistic reasoning, embedding theorem, model-checking

1. Introduction

1.1. Motivations of this Paper

The verification of open, large, randomized, and stochastic concurrent systems is gaining increasing importance in the fields of computer science and engineering. On the one hand, verifying open and large concurrent systems, such as web application systems, requires the use of inconsistency-tolerant (or paraconsistent) reasoning because inconsistencies often appear and are inevitable in such systems [1]. On the other hand, verifying randomized and stochastic concurrent systems, such as fault-tolerant communication systems over unreliable channels, requires the use of probabilistic reasoning because useful notions of reliability for such systems require probabilistic characterization [2]. Thus, the ability to accommodate both inconsistency-tolerant and probabilistic reasoning by an appropriate logic is a requirement for verifying such complex concurrent systems.

Computation tree logic (CTL) [3] is widely accepted as one of the most useful temporal logics for verifying concurrent systems by *model checking* technologies [4]. CTL-based model checking algorithms are known to be more efficient than model-checking algorithms based on other temporal logics such as *linear-time temporal logic* (LTL) [5]. However, CTL is not sufficient for handling paraconsistent and probabilistic accounts of concurrent systems because it has no operators that can represent paraconsistency and probability. Thus, the aim of this paper is to construct a paraconsistent and probabilistic extension of CTL. To achieve this aim, a new logic, *paraconsistent probabilistic CTL* (PpCTL), is introduced. PpCTL is constructed by combining the existing useful CTL-variants, namely *paraconsistent CTL* (PCTL) [6, 7] and *probabilistic CTL* (pCTL) [2, 8] on the basis of a theorem for embedding PpCTL into pCTL. Some illustrative examples describing an *SQL injection* attack detection algorithm [9], which involves the use of PpCTL are also presented in this paper to highlight the virtues of combining paraconsistency (in PCTL) and probability (in pCTL).

Integrating useful reasoning mechanisms is regarded as combining and extending some useful non-classical logics such as *modal logics*. Combining and extending useful non-classical logics are also known to be a very important issue in mathematical logic (see e.g., [10]). The research presented in this paper is thus also intended to overcome this issue and to provide a solution, by combining and extending the following useful non-classical logics: *temporal logic*, *paraconsistent (or inconsistency-tolerant) logic* and *probabilistic (or probability) logic*. Although the proposed embedding-based method is not technically that innovative, it provides a new simple and useful combination of mechanisms for these logics. The combination and extension of these logics enable us to integrate the existing two application areas concerning PCTL and pCTL, respectively.

1.2. Paraconsistent CTL

PCTL, which was introduced and studied by Kamide and Kaneiwa in [6, 7], is a paraconsistent extension of CTL. To appropriately formalize inconsistency-tolerant reasoning, PCTL is based on *Nelson's four-valued paraconsistent logic* N4 [11, 12], which includes a paraconsistent negation connective. The paraconsistent negation connective in PCTL entails the property of *paraconsistency*. Roughly, a satisfaction relation $\models$ is considered to be paraconsistent with respect to a negation connective $\sim$ if the following condition holds:

$$\exists \alpha, \beta\ [M, s \not\models (\alpha \wedge \sim\alpha) \rightarrow \beta],$$

where s is the state of a Kripke structure M. In contrast to PCTL, classical logic has no paraconsistency because the formula of the form

$$(\alpha \wedge \sim\alpha) \rightarrow \beta$$

is valid in classical logic.

Paraconsistent logics, including PCTL, are known to be more appropriate for inconsistency-tolerant and uncertain reasoning than other non-classical logics [13–19]. For example, the following scenario is undesirable:

$$(s(x) \wedge \sim s(x)) \rightarrow d(x)$$

is valid for any symptom s and disease d, where $\sim s(x)$ implies that "a person x does not have a symptom s" and $d(x)$ implies that "a person x suffers from a disease d." An inconsistent scenario expressed as

$$melancholia(john) \wedge \sim melancholia(john)$$

will inevitably occur because melancholia is an uncertain concept and the fact "John has melancholia" may be determined to be true or false by different pathologists with different perspectives. In this case, the undesirable formula

$$(melancholia(john) \wedge \\ \sim melancholia(john)) \rightarrow cancer(john)$$

is valid in classical logic (i.e., an inconsistency has an undesirable consequence), whereas it is not valid in paraconsistent logics (i.e., these logics are inconsistency-tolerant).

We now give a detailed explanation about the usefulness of paraconsistent reasoning. We assume a large medical database *MDB* of symptoms and diseases. We can also assume that *MDB* is inconsistent in the sense that there is a symptom predicate $s(x)$ such that $\sim s(x), s(x) \in MDB$. This assumption is regarded as very realistic, because symptom is an uncertain concept, which is difficult to determine by any diagnosis. It may be determined to be true or false by different doctors with different perspectives. Then, the database *MDB* does not derive arbitrary disease $d(x)$, which means "a person x suffers from a disease d," since paraconsistent logics ensures the fact that for some formulas α and β, the formula $\sim\alpha \wedge \alpha \rightarrow \beta$ is not valid. The paraconsistent logic-based large *MDB* is thus inconsistency-tolerant. In the classical logic, the formula $\sim s(x) \wedge s(x) \rightarrow d(x)$ is valid for any disease d, and hence the non-paraconsistent formulation based on classical logic is regarded as inappropriate for application to this medical database. Apart from such a medical database, large and open concurrent systems also require the handling of paraconsistent scenarios because inconsistencies often appear and are inevitable in these systems. This is the reason why we need to combine PCTL and pCTL.

1.3. Probabilistic CTL

pCTL, which was introduced and studied by Aziz et al. in [8] and Bianco and de Alfaro in [2], is a probabilistic extension of CTL. To appropriately formalize probabilistic reasoning, pCTL uses a *probabilistic* or *probability operator* $\mathrm{P}_{\geq x}$, where the formula of the form $\mathrm{P}_{\geq x}\alpha$ is intended to read "the probability of α holding in the future evolution of the system is at least x." In [2], pCTL and its extension, pCTL*, were introduced for verifying the properties of reliability and the performance of the systems modeled by *discrete Markov chains*. pCTL and pCTL* can appropriately express quantitative bounds on the probability of system evolutions. In addition, in [2], the complexities of model-checking algorithms for pCTL and pCTL* were clarified. In [8], model-checking algorithms for the extensions of the abovementioned settings of pCTL and pCTL* were proposed for verifying probabilistic nondeterministic concurrent systems, in which the probabilistic behavior coexists with nondeterminism. These algorithms were also shown to exhibit polynomial-time complexity depending on the size of the systems.

The main difference between the pCTL settings by Aziz et al. [8] and Bianco and de Alfaro [2] is the setting of the *probability measures* in the *probabilistic Kripke structures* of pCTL. In the present paper, PpCTL is constructed on the basis of a "probability-measure-independent" translation of PpCTL into pCTL. By this translation, a theorem for embedding PpCTL into pCTL is proven, which entails the relative decidability of PpCTL with respect to pCTL, i.e., the decidability of pCTL implies that of PpCTL. This fact indicates that we can reuse the existing pCTL-based verification algorithms by Aziz et al. [8] and Bianco and de Alfaro [2]

The structure of this paper is then summarized as follows.

In Section 2, the new logic PpCTL, which is an extension of both PCTL and pCTL, is introduced on the basis of a *paraconsistent probabilistic Kripke structure* with two types of satisfaction relations. Some remarks on PpCTL are also provided in this section.

In Section 3, a theorem for embedding PpCTL into pCTL is proven using a new translation function. As a corollary of this embedding theorem, a relative decidability theorem for PpCTL, wherein the decidability of pCTL implies that of PpCTL, is obtained. Note that the proposed translation is regarded as a modified extension of the existing translation, which was used by Gurevich [20],

Rautenberg [21], and Vorob'ev [22] to embed *Nelson's three-valued constructive logic* [11, 12] into *positive intuitionistic logic*.

In Section 4, some translation examples are presented.

In Section 5, a bisimulation theorem which is known to be useful for abstraction in model checking is shown for PpCTL.

In Section 6, some illustrative examples for describing the SQL injection attack detection algorithm proposed by Sonoda et al. [9] are presented on the basis of the use of PpCTL-formulas. SQL injection is one of malicious attack methods to exploit security vulnerabilities on SQL database servers.

In Section 7, some remarks on an extension of PpCTL by adding a location operator are addressed.

In Section 8, this paper is concluded and some related works are addressed.

Finally in this section, we remark that this paper includes content that was presented at a conference [23].

2. Logics

Formulas of PpCTL are constructed from countably many atomic formulas, $\rightarrow$ (implication) $\wedge$ (conjunction), $\vee$ (disjunction), $\neg$ (classical negation), $\sim$ (paraconsistent negation), $\mathrm{P}_{\leq x}$ (less than or equal probability), $\mathrm{P}_{\geq x}$ (greater than or equal probability), $\mathrm{P}_{<x}$ (less than probability), $\mathrm{P}_{>x}$ (greater than probability), X (next), G (globally), F (eventually), U (until), R (release), A (all computation paths) and E (some computation path). The symbols X, G, F, U and R are called *temporal operators*, and the symbols A and E are called *path quantifiers*. The symbols $\mathrm{P}_{\leq x}$, $\mathrm{P}_{\geq x}$, $\mathrm{P}_{<x}$ and $\mathrm{P}_{>x}$ are called *probabilistic operators* or *probability operators*. A formula $\mathrm{P}_{\leq x}\alpha$ is intended to read "the probability of α is at least x." The symbol ATOM is used to denote the set of atomic formulas. An expression $A \equiv B$ is used to denote the syntactical identity between A and B.

Definition 2.1: Formulas α are defined by the following grammar, assuming $p \in \mathrm{ATOM}$ and $x \in [0,1]$:

$$\begin{array}{l}\alpha ::= p \mid \alpha{\rightarrow}\alpha \mid \alpha \wedge \alpha \mid \alpha \vee \alpha \mid \neg\alpha \mid {\sim}\alpha \mid \\ \mathrm{P}_{\leq x}\alpha \mid \mathrm{P}_{\geq x}\alpha \mid \mathrm{P}_{<x}\alpha \mid \mathrm{P}_{>x}\alpha \mid \mathrm{AX}\alpha \mid \\ \mathrm{EX}\alpha \mid \mathrm{AG}\alpha \mid \mathrm{EG}\alpha \mid \mathrm{AF}\alpha \mid \mathrm{EF}\alpha \mid \\ \mathrm{A}(\alpha\mathrm{U}\alpha) \mid \mathrm{E}(\alpha\mathrm{U}\alpha) \mid \mathrm{A}(\alpha\mathrm{R}\alpha) \mid \mathrm{E}(\alpha\mathrm{R}\alpha).\end{array}$$

Note that pairs of symbols like AG and EU are indivisible, and that the symbols $\mathrm{X}, \mathrm{G}, \mathrm{F}, \mathrm{U}$ and R cannot occur without being preceded by an A or an E. Similarly, every A or E must have one of X, G, F, U and R to accompany it. It is remarked that all the connectives displayed above are required to obtain a theorem for embedding PpCTL into pCTL.

Definition 2.2: A *paraconsistent probabilistic Kripke structure* (*ppk-structure* for short) is a structure $\langle S, S_0, R, \mu_s, L^+, L^- \rangle$ such that

1. S is the set of states,
2. S_0 is a set of initial states and $S_0 \subseteq S$,
3. R is a binary relation on S which satisfies the condition: $\forall s \in S\ \exists s' \in S\ [(s,s') \in R]$,
4. μ_s is a certain probability measure (or probability distribution) concerning $s \in S$: a set of paths beginning at s is mapped into a real number in $[0,1]$,
5. L^+ and L^- are mappings from S to the power set of a nonempty subset AT of ATOM.

A *path* in a ppk-structure is an infinite sequence of states, $\pi = s_0, s_1, s_2, \ldots$ such that $\forall i \geq 0\ [(s_i, s_{i+1}) \in R]$. The symbol Ω_s is used to denote the set of all paths beginning at s.

Some remarks on the ppk-structure defined above are given as follows.

1. The logic PpCTL will be defined as a ppk-structure with two satisfaction relations $\models^+$ and $\models^-$.
2. The definition of μ_s is not precisely and explicitly given in this paper since the proposed translation from PpCTL into pCTL is independent of the setting of μ_s.
3. There are many possibilities for defining a probability measure μ_s. Some typical examples of probability measures are addressed below.
4. In [2], two probability measures μ_s^+ and μ_s^-, called *minimal probability* and *maximal probability*, respectively, are adopted in pCTL. μ_s^+ and μ_s^- are defined on a Borel σ-algebra $\mathscr{B}_s$ ($\subseteq 2^{\Omega_s}$) as follows: for any $\Delta \in \mathscr{B}_s$, $\mu_s^+(\Delta) = sup\ \mu_{s,\eta}(\Delta)$ and $\mu_s^-(\Delta) = inf\ \mu_{s,\eta}(\Delta)$ where $\mu_{s,\eta}$ with a strategy η concerning nondeterminism is a unique probability measure on $\mathscr{B}_s$.
5. In [8], a probability measure μ^s concerning some *discrete Markov processes* and *discrete generalized Markov processes* is adopted in pCTL. μ^s is defined as a mapping from $\mathscr{C}^s$ into $[0,1]$ where $\mathscr{C}^s$ is a Borel sigma field, which is the class of subsets of the set of all infinite state sequences starting at s.

Definition 2.3—PpCTL: Let AT be a nonempty subset of ATOM. *Satisfaction relations* $\models^+$ and $\models^-$ on a ppk-structure $M = \langle S, S_0, R, \mu_s, L^+, L^- \rangle$ are defined inductively as follows (s represents a state in S):

1. for any $p \in \mathrm{AT}$, $M,s \models^+ p$ iff $p \in L^+(s)$,
2. $M,s \models^+ \alpha_1{\rightarrow}\alpha_2$ iff $M,s \models^+ \alpha_1$ implies $M,s \models^+ \alpha_2$,
3. $M,s \models^+ \alpha_1 \wedge \alpha_2$ iff $M,s \models^+ \alpha_1$ and $M,s \models^+ \alpha_2$,
4. $M,s \models^+ \alpha_1 \vee \alpha_2$ iff $M,s \models^+ \alpha_1$ or $M,s \models^+ \alpha_2$,
5. $M,s \models^+ \neg\alpha_1$ iff $M,s \not\models^+ \alpha_1$,
6. $M,s \models^+ {\sim}\alpha$ iff $M,s \models^- \alpha$,
7. for any $x \in [0,1]$, $M,s \models^+ \mathrm{P}_{\leq x}\alpha$ iff $\mu_s(\{w \in \Omega_s \mid M,s \models^+ \alpha\}) \leq x$,
8. for any $x \in [0,1]$, $M,s \models^+ \mathrm{P}_{\geq x}\alpha$ iff $\mu_s(\{w \in \Omega_s \mid M,s \models^+ \alpha\}) \geq x$,
9. for any $x \in [0,1]$, $M,s \models^+ \mathrm{P}_{<x}\alpha$ iff $\mu_s(\{w \in \Omega_s \mid M,s \models^+ \alpha\}) < x$,

10. for any $x \in [0,1]$, $M,s \models^+ \mathrm{P}_{>x}\alpha$ iff $\mu_s(\{w \in \Omega_s \mid M,s \models^+ \alpha\}) > x$,
11. $M,s \models^+ \mathrm{AX}\alpha$ iff $\forall s_1 \in S\ [(s,s_1) \in R$ implies $M,s_1 \models^+ \alpha]$,
12. $M,s \models^+ \mathrm{EX}\alpha$ iff $\exists s_1 \in S\ [(s,s_1) \in R$ and $M,s_1 \models^+ \alpha]$,
13. $M,s \models^+ \mathrm{AG}\alpha$ iff for all paths $\pi \equiv s_0,s_1,s_2,...$, where $s \equiv s_0$, and all states s_i along π, we have $M,s_i \models^+ \alpha$,
14. $M,s \models^+ \mathrm{EG}\alpha$ iff there is a path $\pi \equiv s_0,s_1,s_2,...$, where $s \equiv s_0$, and for all states s_i along π, we have $M,s_i \models^+ \alpha$,
15. $M,s \models^+ \mathrm{AF}\alpha$ iff for all paths $\pi \equiv s_0,s_1,s_2,...$, where $s \equiv s_0$, there is a state s_i along π such that $M,s_i \models^+ \alpha$,
16. $M,s \models^+ \mathrm{EF}\alpha$ iff there is a path $\pi \equiv s_0,s_1,s_2,...$, where $s \equiv s_0$, and for some state s_i along π, we have $M,s_i \models^+ \alpha$,
17. $M,s \models^+ \mathrm{A}(\alpha_1 \mathrm{U} \alpha_2)$ iff for all paths $\pi \equiv s_0,s_1,s_2,...$, where $s \equiv s_0$, there is a state s_k along π such that $[(M,s_k \models^+ \alpha_2)$ and $\forall j\ (0 \leq j < k$ implies $M,s_j \models^+ \alpha_1)]$,
18. $M,s \models^+ \mathrm{E}(\alpha_1 \mathrm{U} \alpha_2)$ iff there is a path $\pi \equiv s_0,s_1,s_2,...$, where $s \equiv s_0$, and for some state s_k along π, we have $[(M,s_k \models^+ \alpha_2)$ and $\forall j\ (0 \leq j < k$ implies $M,s_j \models^+ \alpha_1)]$,
19. $M,s \models^+ \mathrm{A}(\alpha_1 \mathrm{R} \alpha_2)$ iff for all paths $\pi \equiv s_0,s_1,s_2,...$, where $s \equiv s_0$, and all states s_j along π, we have $[\forall i < j$ not-$[M,s_i \models^+ \alpha_1]$ implies $M,s_j \models^+ \alpha_2]$,
20. $M,s \models^+ \mathrm{E}(\alpha_1 \mathrm{R} \alpha_2)$ iff there is a path $\pi \equiv s_0,s_1,s_2,...$, where $s \equiv s_0$, and for all states s_j along π, we have $[\forall i < j$ not-$[M,s_i \models^+ \alpha_1]$ implies $M,s_j \models^+ \alpha_2]$,
21. for any $p \in \mathrm{AT}$, $M,s \models^- p$ iff $p \in L^-(s)$,
22. $M,s \models^- \alpha_1 \rightarrow \alpha_2$ iff $M,s \models^+ \alpha_1$ and $M,s \models^- \alpha_2$,
23. $M,s \models^- \alpha_1 \wedge \alpha_2$ iff $M,s \models^- \alpha_1$ or $M,s \models^- \alpha_2$,
24. $M,s \models^- \alpha_1 \vee \alpha_2$ iff $M,s \models^- \alpha_1$ and $M,s \models^- \alpha_2$,
25. $M,s \models^- \neg\alpha_1$ iff $M,s \models^+ \alpha_1$,
26. $M,s \models^- \sim\alpha_1$ iff $M,s \models^+ \alpha_1$,
27. for any $x \in [0,1]$, $M,s \models^- \mathrm{P}_{\leq x}\alpha$ iff $\mu_s(\{w \in \Omega_s \mid M,s \models^- \alpha\}) > x$,
28. for any $x \in [0,1]$, $M,s \models^- \mathrm{P}_{\geq x}\alpha$ iff $\mu_s(\{w \in \Omega_s \mid M,s \models^- \alpha\}) < x$,
29. for any $x \in [0,1]$, $M,s \models^- \mathrm{P}_{<x}\alpha$ iff $\mu_s(\{w \in \Omega_s \mid M,s \models^- \alpha\}) \geq x$,
30. for any $x \in [0,1]$, $M,s \models^- \mathrm{P}_{>x}\alpha$ iff $\mu_s(\{w \in \Omega_s \mid M,s \models^- \alpha\}) \leq x$,
31. $M,s \models^- \mathrm{AX}\alpha$ iff $\exists s_1 \in S\ [(s,s_1) \in R$ and $M,s_1 \models^- \alpha]$,
32. $M,s \models^- \mathrm{EX}\alpha$ iff $\forall s_1 \in S\ [(s,s_1) \in R$ implies $M,s_1 \models^- \alpha]$,
33. $M,s \models^- \mathrm{AG}\alpha$ iff there is a path $\pi \equiv s_0,s_1,s_2,...$, where $s \equiv s_0$, and for some state s_i along π, we have $M,s_i \models^- \alpha$,
34. $M,s \models^- \mathrm{EG}\alpha$ iff for all paths $\pi \equiv s_0,s_1,s_2,...$, where $s \equiv s_0$, there is a state s_i along π such that $M,s_i \models^- \alpha$,
35. $M,s \models^- \mathrm{AF}\alpha$ iff there is a path $\pi \equiv s_0,s_1,s_2,...$, where $s \equiv s_0$, and for all states s_i along π, we have $M,s_i \models^- \alpha$,
36. $M,s \models^- \mathrm{EF}\alpha$ iff for all paths $\pi \equiv s_0,s_1,s_2,...$, where $s \equiv s_0$, and all states s_i along π, we have $M,s_i \models^- \alpha$,
37. $M,s \models^- \mathrm{A}(\alpha_1 \mathrm{U} \alpha_2)$ iff there is a path $\pi \equiv s_0,s_1,s_2,...$, where $s \equiv s_0$, and for all states s_j along π, we have $[\forall i < j$ not-$[M,s_i \models^- \alpha_1]$ implies $M,s_j \models^- \alpha_2]$,
38. $M,s \models^- \mathrm{E}(\alpha_1 \mathrm{U} \alpha_2)$ iff for all paths $\pi \equiv s_0,s_1,s_2,...$, where $s \equiv s_0$, and for all states s_j along π, we have $[\forall i < j$ not-$[M,s_i \models^- \alpha_1]$ implies $M,s_j \models^- \alpha_2]$,
39. $M,s \models^- \mathrm{A}(\alpha_1 \mathrm{R} \alpha_2)$ iff there is a path $\pi \equiv s_0,s_1,s_2,...$, where $s \equiv s_0$, and for some state s_k along π, we have $[(M,s_k \models^- \alpha_2)$ and $\forall j\ (0 \leq j < k$ implies $M,s_j \models^- \alpha_1)]$,
40. $M,s \models^- \mathrm{E}(\alpha_1 \mathrm{R} \alpha_2)$ iff for all paths $\pi \equiv s_0,s_1,s_2,...$, where $s \equiv s_0$, there is a state s_k along π such that $[(M,s_k \models^- \alpha_2)$ and $\forall j\ (0 \leq j < k$ implies $M,s_j \models^- \alpha_1)]$.

Definition 2.4: A formula α is *valid* (satisfiable) in PpCTL if $M,s \models^+ \alpha$ holds for any (some) ppk-structure $M = \langle S,S_0,R,\mu_s,\ L^+,L^-\rangle$, any (some) $s \in S$, and any (some) satisfaction relations $\models^+$ and $\models^-$ on M.

Definition 2.5: Let M be a ppk-structure $\langle S,S_0,R,\mu_s,\ L^+,L^-\rangle$ for PpCTL, and $\models^+$ and $\models^-$ be satisfaction relations on M. Then, the *positive and negative model checking problems* for PpCTL are respectively defined by: for any formula α, find the sets $\{s \in S \mid M,s \models^+ \alpha\}$ and $\{s \in S \mid M,s \models^- \alpha\}$.

Some remarks on PpCTL are then given as follows.

1. The intuitive meanings of $\models^+$ and $\models^-$ in PpCTL are "verification (or justification)" and "refutation (or falsification)," respectively [15, 16].
2. PpCTL is regarded as a four-valued logic, since for each $s \in S$ and each formula α, we can take one of the following four cases:
   a. α is verified at s, i.e., $M,s \models^+ \alpha$,
   b. α is falsified at s, i.e., $M,s \models^- \alpha$,
   c. α is both verified and falsified at s,
   d. α is neither verified nor falsified at s.
3. PpCTL is regarded as a paraconsistent logic. This is explained as follows. Assume a ppk-structure $M = \langle S,S_0,R,\mu_s,L^+,L^-\rangle$ such that $p \in L^+(s)$, $p \in L^-(s)$ and $q \notin L^+(s)$ for any distinct atomic formulas p and q. Then, $M,s \models^+ (p \wedge \sim p) \rightarrow q$ does not hold, and hence $\models^+$ in PpCTL is paraconsistent with respect to $\sim$. For more information on paraconsistency, see e.g., [13].
4. The positive model checking problem for PpCTL corresponds to the standard "verification-based" model checking problem for pCTL. The negative model checking problem for PpCTL corresponds to the dual of positive one. i.e., it is regarded as a "refutation-based" model checking problem. Both the positive and negative model checking should simultaneously be performed, i.e., only one of them cannot be performed.

An expression $\alpha \leftrightarrow \beta$ is used to represent $(\alpha \rightarrow \beta) \wedge (\beta \rightarrow \alpha)$.

Proposition 2.6: The following formulas concerning paraconsistent negation are valid in PpCTL: for any formulas α and β,

1. $\sim\sim\alpha \leftrightarrow \alpha$,
2. $\sim\neg\alpha \leftrightarrow \alpha$,
3. $\sim(\alpha \wedge \beta) \leftrightarrow \sim\alpha \vee \sim\beta$,

4. $\sim(\alpha \vee \beta) \leftrightarrow \sim\alpha \wedge \sim\beta$,
5. $\sim(\alpha \rightarrow \beta) \leftrightarrow \alpha \wedge \sim\beta$,
6. $\sim \mathrm{P}_{\leq x}\alpha \leftrightarrow \mathrm{P}_{>x}\sim\alpha$,
7. $\sim \mathrm{P}_{\geq x}\alpha \leftrightarrow \mathrm{P}_{<x}\sim\alpha$,
8. $\sim \mathrm{P}_{<x}\alpha \leftrightarrow \mathrm{P}_{\geq x}\sim\alpha$,
9. $\sim \mathrm{P}_{>x}\alpha \leftrightarrow \mathrm{P}_{\leq x}\sim\alpha$,
10. $\sim \mathrm{AX}\alpha \leftrightarrow \mathrm{EX}\sim\alpha$,
11. $\sim \mathrm{EX}\alpha \leftrightarrow \mathrm{AX}\sim\alpha$,
12. $\sim \mathrm{AG}\alpha \leftrightarrow \mathrm{EF}\sim\alpha$,
13. $\sim \mathrm{EG}\alpha \leftrightarrow \mathrm{AF}\sim\alpha$,
14. $\sim \mathrm{AF}\alpha \leftrightarrow \mathrm{EG}\sim\alpha$,
15. $\sim \mathrm{EF}\alpha \leftrightarrow \mathrm{AG}\sim\alpha$,
16. $\sim \mathrm{A}(\alpha \mathrm{U} \beta) \leftrightarrow \mathrm{E}((\sim\alpha)\mathrm{R}(\sim\beta))$,
17. $\sim \mathrm{E}(\alpha \mathrm{U} \beta) \leftrightarrow \mathrm{A}((\sim\alpha)\mathrm{R}(\sim\beta))$,
18. $\sim \mathrm{A}(\alpha \mathrm{R} \beta) \leftrightarrow \mathrm{E}((\sim\alpha)\mathrm{U}(\sim\beta))$,
19. $\sim \mathrm{E}(\alpha \mathrm{R} \beta) \leftrightarrow \mathrm{A}((\sim\alpha)\mathrm{U}(\sim\beta))$.

Proof. We show some characteristic cases. Suppose that $M = \langle S, S_0, R, \mu_s, L^+, L^- \rangle$ is an arbitrary ppk-structure, and that $\models^+$ and $\models^-$ are any satisfaction relations on M.

(1): We show that $\sim\sim\alpha \rightarrow \alpha$ is valid in PpCTL. Let s be an arbitrary element of S. Then, to show $M, s \models^+ \sim\sim\alpha \rightarrow \alpha$, we show that $M, s \models^+ \sim\sim\alpha$ implies $M, s \models^+ \alpha$. Suppose $M, \models^+ \sim\sim\alpha$. Then, we obtain the required fact as follows: $M, s \models^+ \sim\sim\alpha$ iff $M, s \models^- \sim\alpha$ iff $M, s \models^+ \alpha$. In a similar way, we can also show that $\alpha \rightarrow \sim\sim\alpha$ is valid in PpCTL.

(2): We show that $\sim\neg\alpha \rightarrow \alpha$ is valid in PpCTL. Let s be an arbitrary element of S. Then, to show $M, s \models^+ \sim\neg\alpha \rightarrow \alpha$, we show that $M, s \models^+ \sim\neg\alpha$ implies $M, s \models^+ \alpha$. Suppose $M, \models^+ \sim\neg\alpha$. Then, we obtain the required fact as follows: $M, s \models^+ \sim\neg\alpha$ iff $M, s \models^- \neg\alpha$ iff $M, s \models^+ \alpha$. In a similar way, we can also show that $\alpha \rightarrow \sim\neg\alpha$ is valid in PpCTL.

(6): We show that $\sim \mathrm{P}_{\leq x}\alpha \rightarrow \mathrm{P}_{>x}\sim\alpha$ is valid in PpCTL. Let s be an arbitrary element of S. Then, we show $M, s \models^+ \sim \mathrm{P}_{\leq x}\alpha \rightarrow \mathrm{P}_{>x}\sim\alpha$. To show this, we show that $M, s \models^+ \sim \mathrm{P}_{\leq x}\alpha$ implies $M, s \models^+ \mathrm{P}_{>x}\sim\alpha$. Suppose $M, \models^+ \sim \mathrm{P}_{\leq x}\alpha$. Then, we obtain the required fact as follows: $M, \models^+ \sim \mathrm{P}_{\leq x}\alpha$ iff $M, \models^- \mathrm{P}_{\leq x}\alpha$ iff $\mu_s(\{w \in \Omega_s \mid M, w \models^- \alpha\}) > x$ iff $\mu_s(\{w \in \Omega_s \mid M, w \models^+ \sim\alpha\}) > x$ iff $M, s \models^+ \mathrm{P}_{>x}\sim\alpha$. In a similar way, we can also show that $\mathrm{P}_{>x}\sim\alpha \rightarrow \sim \mathrm{P}_{\leq x}\alpha$ is valid in PpCTL.

(9): We show that $\sim \mathrm{P}_{>x}\alpha \rightarrow \mathrm{P}_{\leq x}\sim\alpha$ is valid in PpCTL. Let s be an arbitrary element of S. Then, we show $M, s \models^+ \sim \mathrm{P}_{>x}\alpha \rightarrow \mathrm{P}_{\leq x}\sim\alpha$. To show this, we show that $M, s \models^+ \sim \mathrm{P}_{>x}\alpha$ implies $M, s \models^+ \mathrm{P}_{\leq x}\sim\alpha$. Suppose $M, \models^+ \sim \mathrm{P}_{>x}\alpha$. Then, we obtain the required fact as follows: $M, \models^+ \sim \mathrm{P}_{>x}\alpha$ iff $M, \models^- \mathrm{P}_{>x}\alpha$ iff $\mu_s(\{w \in \Omega_s \mid M, w \models^- \alpha\}) \leq x$ iff $\mu_s(\{w \in \Omega_s \mid M, w \models^+ \sim\alpha\}) \leq x$ iff $M, s \models^+ \mathrm{P}_{\leq x}\sim\alpha$. In a similar way, we can also show that $\mathrm{P}_{\leq x}\sim\alpha \rightarrow \sim \mathrm{P}_{>x}\alpha$ is valid in PpCTL.

(16): We show that $\sim \mathrm{A}(\alpha \mathrm{U} \beta) \rightarrow \mathrm{E}((\sim\alpha)\mathrm{R}(\sim\beta))$ is valid in PpCTL. Let s be an arbitrary element of S. Then, we show that $M, s \models^+ \sim \mathrm{A}(\alpha \mathrm{U} \beta) \rightarrow \mathrm{E}((\sim\alpha)\mathrm{R}(\sim\beta))$. To show this, we show that $M, s \models^+ \sim \mathrm{A}(\alpha \mathrm{U} \beta)$ implies $M, s \models^+ \mathrm{E}((\sim\alpha)\mathrm{R}(\sim\beta))$. Suppose $M, s \models^+ \sim \mathrm{A}(\alpha \mathrm{U} \beta)$, i.e., $M, s \models^- \mathrm{A}(\alpha \mathrm{U} \beta)$. Then, we obtain the required fact as follows:

$M, s \models^- \mathrm{A}(\alpha \mathrm{U} \beta)$

iff there is a path $\pi \equiv s_0, s_1, s_2, \ldots$, where $s \equiv s_0$, and for all states s_j along π, we have $[\forall i < j$ not-$[M, s_i \models^- \alpha]$ implies $M, s_j \models^- \beta]$

iff there is a path $\pi \equiv s_0, s_1, s_2, \ldots$, where $s \equiv s_0$, and for all states s_j along π, we have $[\forall i < j$ not-$[N, s_i \models^+ \sim\alpha]$ implies $N, s_j \models^+ \sim\beta]$

iff $M, s \models^+ \mathrm{E}((\sim\alpha)\mathrm{R}(\sim\beta))$.

In a similar way, we can also show that $\mathrm{E}((\sim\alpha)\mathrm{R}(\sim\beta)) \rightarrow \sim \mathrm{A}(\alpha \mathrm{U} \beta))$ is valid in PpCTL.

■

In order to define a translation, the logic pCTL is defined below.

Definition 2.7—pCTL: A *probabilistic Kripke structure* (*pk-structure* for short) for pCTL is a structure $\langle S, S_0, R, \mu_s, L \rangle$ such that

1. S, S_0, R and μ_s are the same as those in Definition 2.2,
2. L is a mapping from S to the power set of a nonempty subset AT of ATOM.

A *satisfaction relation* $\models$ on a pk-structure $M = \langle S, S_0, R, \mu, L \rangle$ for pCTL is defined by the same conditions for $\models^+$ (except the condition 6) as in Definition 2.3 (by deleting the superscript $+$). The validity, satisfiability and model-checking problems for pCTL are defined similarly as those for PpCTL.

It is remarked that $\models^+$ of PpCTL includes $\models$ of pCTL, and hence PpCTL is an extension of pCTL.

3. Embeddability and Relative Decidability

In this section, we introduce a translation function from PpCTL into pCTL. By using this translation function, we show the embedding theorem of PpCTL into pCTL and the relative decidability theorem for PpCTL.

Definition 3.1: Let AT be a non-empty subset of ATOM, and AT′ be the set $\{p' \mid p \in \mathrm{AT}\}$ of atomic formulas. The language $\mathscr{L}^{\sim}$ (the set of formulas) of PpCTL is defined using AT, $\sim$, $\rightarrow, \wedge, \vee, \neg$, $\mathrm{P}_{\leq x}, \mathrm{P}_{\geq x}, \mathrm{P}_{<x}, \mathrm{P}_{>x}$, X, F, G, U, R, A and E. The language $\mathscr{L}$ of pCTL is obtained from $\mathscr{L}^{\sim}$ by adding AT′ and deleting $\sim$.

A mapping f from $\mathscr{L}^{\sim}$ to $\mathscr{L}$ is defined inductively by:

1. for any $p \in \mathrm{AT}$, $f(p) := p$ and $f(\sim p) := p' \in \mathrm{AT}'$,
2. $f(\alpha \sharp \beta) := f(\alpha) \sharp f(\beta)$ where $\sharp \in \{\wedge, \vee, \rightarrow\}$,
3. $f(\sharp\alpha) := \sharp f(\alpha)$ where $\sharp \in \{\neg, \mathrm{P}_{\leq x}, \mathrm{P}_{\geq x}, \mathrm{P}_{<x}, \mathrm{P}_{>x}, \mathrm{AX}, \mathrm{EX}, \mathrm{AG}, \mathrm{EG}, \mathrm{AF}, \mathrm{EF}\}$,
4. $f(\mathrm{A}(\alpha \mathrm{U} \beta))) := \mathrm{A}(f(\alpha) \mathrm{U} f(\beta))$,
5. $f(\mathrm{E}(\alpha \mathrm{U} \beta))) := \mathrm{E}(f(\alpha) \mathrm{U} f(\beta))$,
6. $f(\mathrm{A}(\alpha \mathrm{R} \beta))) := \mathrm{A}(f(\alpha) \mathrm{R} f(\beta))$,
7. $f(\mathrm{E}(\alpha \mathrm{R} \beta))) := \mathrm{E}(f(\alpha) \mathrm{R} f(\beta))$,
8. $f(\sim\sim\alpha) := f(\alpha)$,
9. $f(\sim(\alpha \rightarrow \beta)) := f(\alpha) \wedge f(\sim\beta)$,
10. $f(\sim(\alpha \wedge \beta)) := f(\sim\alpha) \vee f(\sim\beta)$,
11. $f(\sim(\alpha \vee \beta)) := f(\sim\alpha) \wedge f(\sim\beta)$,
12. $f(\sim\neg\alpha) := f(\alpha)$,
13. $f(\sim \mathrm{P}_{\leq x}\alpha) := \mathrm{P}_{>x} f(\sim\alpha)$,
14. $f(\sim \mathrm{P}_{\geq x}\alpha) := \mathrm{P}_{<x} f(\sim\alpha)$,
15. $f(\sim \mathrm{P}_{<x}\alpha) := \mathrm{P}_{\geq x} f(\sim\alpha)$,
16. $f(\sim \mathrm{P}_{>x}\alpha) := \mathrm{P}_{\leq x} f(\sim\alpha)$,
17. $f(\sim \mathrm{AX}\alpha) := \mathrm{EX} f(\sim\alpha)$,
18. $f(\sim \mathrm{EX}\alpha) := \mathrm{AX} f(\sim\alpha)$,
19. $f(\sim \mathrm{AG}\alpha) := \mathrm{EF} f(\sim\alpha)$,
20. $f(\sim \mathrm{EG}\alpha) := \mathrm{AF} f(\sim\alpha)$,
21. $f(\sim \mathrm{AF}\alpha) := \mathrm{EG} f(\sim\alpha)$,
22. $f(\sim \mathrm{EF}\alpha) := \mathrm{AG} f(\sim\alpha)$,
23. $f(\sim(\mathrm{A}(\alpha \mathrm{U} \beta))) := \mathrm{E}(f(\sim\alpha) \mathrm{R} f(\sim\beta))$,
24. $f(\sim(\mathrm{E}(\alpha \mathrm{U} \beta))) := \mathrm{A}(f(\sim\alpha) \mathrm{R} f(\sim\beta))$,
25. $f(\sim(\mathrm{A}(\alpha \mathrm{R} \beta))) := \mathrm{E}(f(\sim\alpha) \mathrm{U} f(\sim\beta))$,
26. $f(\sim(\mathrm{E}(\alpha \mathrm{R} \beta))) := \mathrm{A}(f(\sim\alpha) \mathrm{U} f(\sim\beta))$.

In order to show the embedding theorem, we need to show some lemmas.

Lemma 3.2: Let f be the mapping defined in Definition 3.1. For any ppk-structure $M := \langle S, S_0, R, \mu_s, L^+, L^- \rangle$ for PpCTL, and any satisfaction relations $\models^+$ and $\models^-$ on M, we can construct a pk-structure $N := \langle S, S_0, R, \mu_s, L \rangle$ for CTL and a satisfaction relation $\models$ on N such that for any formula α in $\mathscr{L}^{\sim}$ and any state s in S,

1. $M, s \models^+ \alpha$ iff $N, s \models f(\alpha)$,
2. $M, s \models^- \alpha$ iff $N, s \models f(\sim\alpha)$.

Proof. Let AT be a nonempty subset of ATOM, and AT′ be the set $\{p' \mid p \in \mathrm{AT}\}$ of atomic formulas. Suppose that M is a ppk-structure $\langle S, S_0, R, \mu_s, L^+, L^- \rangle$ such that

> L^+ and L^- are mappings from S to the power set of AT.

Suppose that N is a pk-structure $M := \langle S, S_0, R, \mu_s, L \rangle$ such that

> L is a mapping from S to the power set of AT $\cup$ AT′.

Suppose moreover that for any $s \in S$ and any $p \in \mathrm{AT}$,

1. $p \in L^+(s)$ iff $p \in L(s)$,
2. $p \in L^-(s)$ iff $p' \in L(s)$.

The lemma is then proved by (simultaneous) induction on the complexity of α.

• Base step:

> Case $\alpha \equiv p \in \mathrm{AT}$: For (1), we obtain: $M, s \models^+ p$ iff $p \in L^+(s)$ iff $p \in L(s)$ iff $N, s \models p$ iff $N, s \models f(p)$ (by the definition of f). For (2), we obtain: $M, s \models^- p$ iff $p \in L^-(s)$ iff $p' \in L(s)$ iff $N, s \models p'$ iff $N, s \models f(\sim p)$ (by the definition of f).

• Induction step: We show some cases.

1. Case $\alpha \equiv \beta \wedge \gamma$: For (1), we obtain: $M, s \models^+ \beta \wedge \gamma$ iff $M, s \models^+ \beta$ and $M, s \models^+ \gamma$ iff $N, s \models f(\beta)$ and $N, s \models f(\gamma)$ (by induction hypothesis for 1) iff $N, s \models f(\beta) \wedge f(\gamma)$ iff $N, s \models f(\beta \wedge \gamma)$ (by the definition of f). For (2), we obtain: $M, s \models^- \beta \wedge \gamma$ iff $M, s \models^- \beta$ or $M, s \models^- \gamma$ iff $N, s \models f(\sim\beta)$ or $N, s \models f(\sim\gamma)$ (by induction hypothesis for 2) iff $N, s \models f(\sim\beta) \vee f(\sim\gamma)$ iff $N, s \models f(\sim(\beta \wedge \gamma))$ (by the definition of f).
2. Case $\alpha \equiv \beta \rightarrow \gamma$: For (1), we obtain: $M, s \models^+ \beta \rightarrow \gamma$ iff $M, s \models^+ \beta$ implies $M, s \models^+ \gamma$ iff $N, s \models f(\beta)$ implies $N, s \models f(\gamma)$ (by induction hypothesis for 1) iff $N, s \models f(\beta) \rightarrow f(\gamma)$ iff $N, s \models f(\beta \rightarrow \gamma)$ (by the definition of f). For (2), we obtain: $M, s \models^- \beta \rightarrow \gamma$ iff $M, s \models^+ \beta$ and $M, s \models^- \gamma$ iff $N, s \models f(\beta)$ and $N, s \models f(\sim\gamma)$ (by induction hypothesis for 1 and 2) iff $N, s \models f(\beta) \wedge f(\sim\gamma)$ iff $N, s \models f(\sim(\beta \rightarrow \gamma))$ (by the definition of f).
3. Case $\alpha \equiv \neg\beta$: For (1), we obtain: $M, s \models^+ \neg\beta$ iff $M, s \not\models^+ \beta$ iff $[N, s \not\models f(\beta)$ (by induction hypothesis for 1) iff $N, s \models \neg f(\beta)$ iff $N, s \models f(\neg\beta)$ (by the definition of f). For (2), we obtain: $M, s \models^- \neg\beta$ iff $M, s \models^+ \beta$ iff $N, s \models f(\beta)$ (by induction hypothesis for 1) iff $N, s \models f(\sim\neg\beta)$ (by the definition of f).
4. Case $\alpha \equiv \sim\beta$: For (1), we obtain: $M, s \models^+ \sim\beta$ iff $M, s \models^- \beta$ iff $N, s \models f(\sim\beta)$ (by induction hypothesis for 2). For (2), we obtain: $M, s \models^- \sim\beta$ iff $M, s \models^+ \beta$ iff $N, s \models f(\beta)$ (by induction hypothesis for 1) iff $N, s \models f(\sim\sim\beta)$ (by the definition of f).

5. Case $\alpha \equiv \mathrm{P}_{\leq x}\beta$: For (1), we obtain: $M,s \models^{+} \mathrm{P}_{\leq x}\beta$ iff $\mu_s(\{w \in \Omega_s \mid M,w \models^{+} \beta\}) \leq x$ iff $\mu_s(\{w \in \Omega_s \mid N,w \models f(\beta)\}) \leq x$ (by induction hypothesis for 1) iff $N,s \models \mathrm{P}_{\leq x} f(\beta)$ iff $N,s \models f(\mathrm{P}_{\leq x}\beta)$ (by the definition of f). For (2), we obtain: $M,s \models^{-} \mathrm{P}_{\leq x}\beta$ iff $\mu_s(\{w \in \Omega_s \mid M,w \models^{-} \beta\}) > x$ iff $\mu_s(\{w \in \Omega_s \mid N,w \models f(\sim\beta)\}) > x$ (by induction hypothesis for 2) iff $N,s \models \mathrm{P}_{>x} f(\sim\beta)$ iff $N,s \models f(\sim\mathrm{P}_{\leq x}\beta)$ (by the definition of f).

6. Case $\alpha \equiv \mathrm{P}_{<x}\beta$: For (1), we obtain: $M,s \models^{+} \mathrm{P}_{<x}\beta$ iff $\mu_s(\{w \in \Omega_s \mid M,w \models^{+} \beta\}) < x$ iff $\mu_s(\{w \in \Omega_s \mid N,w \models f(\beta)\}) < x$ (by induction hypothesis for 1) iff $N,s \models \mathrm{P}_{<x} f(\beta)$ iff $N,s \models f(\mathrm{P}_{<x}\beta)$ (by the definition of f). For (2), we obtain: $M,s \models^{-} \mathrm{P}_{<x}\beta$ iff $\mu_s(\{w \in \Omega_s \mid M,w \models^{-} \beta\}) \geq x$ iff $\mu_s(\{w \in \Omega_s \mid N,w \models f(\sim\beta)\}) \geq x$ (by induction hypothesis for 2) iff $N,s \models \mathrm{P}_{\geq x} f(\sim\beta)$ iff $N,s \models f(\sim\mathrm{P}_{<x}\beta)$ (by the definition of f).

7. Case $\alpha \equiv \mathrm{AX}\beta$: For (1), we obtain: $M,s \models^{+} \mathrm{AX}\beta$ iff $\forall s_1 \in S\,[(s,s_1) \in R$ implies $M,s_1 \models^{+} \beta]$ iff $\forall s_1 \in S\,[(s,s_1) \in R$ implies $N,s_1 \models f(\beta)]$ (by induction hypothesis for 1) iff $N,s \models \mathrm{AX} f(\beta)$ iff $N,s \models f(\mathrm{AX}\beta)$ (by the definition of f). For (2), we obtain: $M,s \models^{-} \mathrm{AX}\beta$ iff $\exists s_1 \in S\,[(s,s_1) \in R$ and $M,s_1 \models^{-} \beta]$ iff $\exists s_1 \in S\,[(s,s_1) \in R$ and $N,s_1 \models f(\sim\beta)]$ (by induction hypothesis for 2) iff $N,s \models \mathrm{EX} f(\sim\beta)$ iff $N,s \models f(\sim\mathrm{AX}\beta)$ (by the definition of f).

8. Case $\alpha \equiv \mathrm{AG}\beta$: For (1), we obtain:

   $M,s \models^{+} \mathrm{AG}\beta$

   iff for all paths $\pi \equiv s_0,s_1,s_2,...$, where $s \equiv s_0$, and all states s_i along π, we have $M,s_i \models^{+} \beta$

   iff for all paths $\pi \equiv s_0,s_1,s_2,...$, where $s \equiv s_0$, and all states s_i along π, we have $N,s_i \models f(\beta)$ (by induction hypothesis for 1)

   iff $N,s \models \mathrm{AG} f(\beta)$

   iff $N,s \models f(\mathrm{AG}\beta)$ (by the definition of f).

   For (2), we obtain:

   $M,s \models^{-} \mathrm{AG}\beta$

   iff there is a path $\pi \equiv s_0,s_1,s_2,...$, where $s \equiv s_0$, for some state s_i along π, we have $M,s_i \models^{-} \beta$

   iff there is a path $\pi \equiv s_0,s_1,s_2,...$, where $s \equiv s_0$, for some state s_i along π, we have $N,s_i \models f(\sim\beta)$ (by induction hypothesis for 2)

   iff $N,s \models \mathrm{EF} f(\sim\beta)$

   iff $N,s \models f(\sim\mathrm{AG}\beta))$ (by the definition of f).

9. Case $\alpha \equiv \mathrm{A}(\beta\mathrm{U}\gamma)$: For (1), we obtain:

   $M,s \models^{+} \mathrm{A}(\beta\mathrm{U}\gamma)$

   iff for all paths $\pi \equiv s_0,s_1,s_2,...$, where $s \equiv s_0$, there is a state s_k along π such that $[M,s_k \models^{+} \gamma$ and $\forall j[i \leq j < k$ implies $M,s_j \models^{+} \beta]$

   iff for all paths $\pi \equiv s_0,s_1,s_2,...$, where $s \equiv s_0$, there is a state s_k along π such that $[N,s_k \models f(\gamma)$ and $\forall j[i \leq j < k$ implies $N,s_j \models f(\beta)]$ (by induction hypothesis for 1)

   iff $N,s \models \mathrm{A}(f(\beta)\mathrm{U}f(\gamma))$

   iff $N,s \models f(\mathrm{A}(\beta\mathrm{U}\gamma))$ (by the definition of f).

   For (2), we obtain:

   $M,s \models^{-} \mathrm{A}(\beta\mathrm{U}\gamma)$

   iff there is a path $\pi \equiv s_0,s_1,s_2,...$, where $s \equiv s_0$, and for all states s_j along π, we have $[\forall i < j$ not-$[M,s_i \models^{-} \beta]$ implies $M,s_j \models^{-} \gamma]$

   iff there is a path $\pi \equiv s_0,s_1,s_2,...$, where $s \equiv s_0$, and for all states s_j along π, we have $[\forall i < j$ not-$[N,s_i \models f(\sim\beta)]$ implies $N,s_j \models f(\sim\gamma)]$ (by induction hypothesis for 2)

   iff $N,s \models \mathrm{E}(f(\sim\beta)\mathrm{R}f(\sim\gamma))$

   iff $N,s \models f(\sim(\mathrm{A}(\beta\mathrm{U}\gamma)))$ (by the definition of f).

■

Lemma 3.3: Let f be the mapping defined in Definition 3.1. For any pk-structure $N := \langle S,S_0,R,\mu_s,L\rangle$ for pCTL, and any satisfaction relation $\models$ on N, we can construct a ppk-structure $M := \langle S,S_0,R,\mu_s,L^{+},L^{-}\rangle$ for PpCTL and satisfaction relations $\models^{+}$ and $\models^{-}$ on M such that for any formula α in $\mathscr{L}^{\sim}$ and any state s in S,

1. $N,s \models f(\alpha)$ iff $M,s \models^{+} \alpha$,
2. $N,s \models f(\sim\alpha)$ iff $M,s \models^{-} \alpha$.

Proof. Similar to the proof of Lemma 3.2. ■

By using Lemmas 3.2 and 3.3, we obtain the following embedding and relative decidability theorems.

Theorem 3.4—Embeddability: Let f be the mapping defined in Definition 3.1. For any formula α,

> α is valid (satisfiable) in PpCTL iff $f(\alpha)$ is valid (satisfiable, resp.) in pCTL.

Proof. By Lemmas 3.2 and 3.3. ■

Theorem 3.5—Relative decidability: If the model-checking, validity and satisfiability problems for pCTL with a probability measure are decidable, then the model-checking, validity and satisfiability problems for PpCTL with the same probability measure as that of pCTL are also decidable.

Proof. Suppose that the probability measure μ_s in the underlying ppk-structure $\langle S,S_0,R,\ \mu_s,\ L^{+},L^{-}\rangle$ of PpCTL is the same as the underlying pk-structure $\langle S,S_0,R,\ \mu_s,\ L\rangle$ of pCTL. Suppose also that pCTL with μ_s is decidable. Then, by the mapping f defined in Definition 3.1, a formula α of PpCTL can be transformed into the corresponding formula $f(\alpha)$ of pCTL. By Lemmas 3.2 and 3.3 and Theorem 3.4, the model checking, validity and satisfiability problems for PpCTL can be transformed into those of pCTL. Since the model checking, validity and satisfiability problems for pCTL with μ_s are decidable by the assumption, the problems for PpCTL with μ_s are also decidable. ■

Some remarks on the decidability of PpCTL are given as follows.

1. The logic pCTL with two probability measures μ_s^+ and μ_s^- by Bianco and de Alfaro is decidable [2]. The logic pCTL with a probability measure μ^s by Aziz et al. is also decidable [8]. Thus, the extended PpCTLs based on the above pCTLs are also decidable by Corollary 3.5.

2. Since the mapping f from PpCTL into pCTL is a polynomial-time reduction, the complexity results for PpCTL become the same results as those for pCTL., e.g., if the model-checking problem for pCTL is deterministic PTIME-complete, then so is PpCTL.

3. The model-checking, validity and satisfiability problems for both CTL and its paraconsistent extension PCTL [7] are known to be EXPTIME-complete, deterministic EXPTIME-complete and deterministic PTIME-complete, respectively.

4. Translation Examples

We provide an algorithm for translating a PpCTL-formula into a pCTL-formula.

Algorithm 4.1: Let α be a PpCTL-formula. Then, we translate the PpCTL-formula α into a pCTL-formula $f(\alpha)$ by using the translation function f defined in Definition 3.1.

1. For α, we apply the condition of f which corresponds to the outer-most PpCTL-connective of α.

2. The resulting formula expression β is of the form $f(\alpha_1) \circ f(\alpha_2)$, $\sharp f(\alpha_1)$, $\sharp(f(\alpha_1) \circ f(\alpha_2))$ or $f(\alpha_1)$ where $\circ$ and $\sharp$ represent the outer-most PpCTL-connectives of α.

3. If there is a PpCTL-connective appearing in α_1 and/or α_2 in β, then we apply the same procedure displayed above to α_1 and α_2, where α_1 and/or α_2 are regarded as α above. If there is no PpCTL-connective appearing in α_1 and/or α_2 in β, then we go to the next step.

4. We translate all the formulas of the form $\sim p$ appearing in the resulting formula expression into the pCTL-formulas of the form p'.

5. The resulting formula expression is just a required pCTL-formula $f(\alpha)$.

We now show some translation examples based on the above algorithm.

Example 4.2: We consider a formula $\mathrm{P}_{>x}{\sim}p{\rightarrow}{\sim}\mathrm{P}_{\leq x}p$ where p is an atomic formula. We translate this PpCTL-formula into a pCTL-formula by the translation function f as follows:

$$
\begin{aligned}
&f(\mathrm{P}_{>x}{\sim}p{\rightarrow}{\sim}\mathrm{P}_{\leq x}p)\\
&= f(\mathrm{P}_{>x}{\sim}p){\rightarrow}f({\sim}\mathrm{P}_{\leq x}p)\\
&= \mathrm{P}_{>x}f({\sim}p){\rightarrow}\mathrm{P}_{>x}f({\sim}p)\\
&= \mathrm{P}_{>x}p'{\rightarrow}\mathrm{P}_{>x}p'
\end{aligned}
$$

where p' is an atomic formula in pCTL. Thus, the formula $\mathrm{P}_{>x}{\sim}p{\rightarrow}{\sim}\mathrm{P}_{\leq x}p$ in PpCTL is translated into the formula $\mathrm{P}_{>x}p'{\rightarrow}\mathrm{P}_{>x}p'$ in pCTL.

Example 4.3: We consider a formula $\mathrm{P}_{\leq x}({\sim}\mathrm{A}(p\mathrm{U}q)){\rightarrow}\mathrm{A}(({\sim}\mathrm{P}_{>x}p)\mathrm{U}({\sim}\mathrm{P}_{>x}q))$ where p and q are atomic formulas. We translate this PpCTL-formula into a pCTL-formula by the translation function f as follows:

$$
\begin{aligned}
&f(\mathrm{P}_{\leq x}({\sim}\mathrm{A}(p\mathrm{U}q)){\rightarrow}\mathrm{A}(({\sim}\mathrm{P}_{>x}p)\mathrm{U}({\sim}\mathrm{P}_{>x}q)))\\
&= f(\mathrm{P}_{\leq x}({\sim}\mathrm{A}(p\mathrm{U}q))){\rightarrow}f(\mathrm{A}(({\sim}\mathrm{P}_{>x}p)\mathrm{U}({\sim}\mathrm{P}_{>x}q)))\\
&= \mathrm{P}_{\leq x}f({\sim}\mathrm{A}(p\mathrm{U}q)){\rightarrow}\mathrm{A}(f({\sim}\mathrm{P}_{>x}p)\mathrm{U}f({\sim}\mathrm{P}_{>x}q))\\
&= \mathrm{P}_{\leq x}\mathrm{E}(f({\sim}p)\mathrm{R}f({\sim}q)){\rightarrow}\mathrm{A}(f({\sim}\mathrm{P}_{>x}p)\mathrm{U}f({\sim}\mathrm{P}_{>x}q))\\
&= \mathrm{P}_{\leq x}\mathrm{E}(p'\mathrm{R}q'){\rightarrow}\mathrm{A}((\mathrm{P}_{\leq x}f({\sim}p))\mathrm{U}(\mathrm{P}_{\leq x}f({\sim}q)))\\
&= \mathrm{P}_{\leq x}\mathrm{E}(p'\mathrm{R}q'){\rightarrow}\mathrm{A}((\mathrm{P}_{\leq x}p')\mathrm{U}(\mathrm{P}_{\leq x}q'))
\end{aligned}
$$

where p', q' are atomic formulas in pCTL. Thus, the formula $\mathrm{P}_{\leq x}({\sim}\mathrm{A}(p\mathrm{U}q)) \rightarrow \mathrm{A}(({\sim}\mathrm{P}_{>x}p)\mathrm{U}({\sim}\mathrm{P}_{>x}q))$ in PpCTL is translated into the formula $\mathrm{P}_{\leq x}\mathrm{E}(p'\mathrm{R}q') \rightarrow \mathrm{A}((\mathrm{P}_{\leq x}p')\mathrm{U}(\mathrm{P}_{\leq x}q'))$ in pCTL.

Example 4.4: We consider a formula ${\sim}\mathrm{AG}(\mathrm{P}_{\leq x}p \wedge {\sim}\mathrm{P}_{>x}q{\rightarrow}\mathrm{EF}r)$ where p, q and r are atomic formulas. We translate this PpCTL-formula into a pCTL-formula by the translation function f as follows:

$$
\begin{aligned}
&f({\sim}\mathrm{AG}(\mathrm{P}_{\leq x}p \wedge {\sim}\mathrm{P}_{>x}q{\rightarrow}\mathrm{EF}r))\\
&= \mathrm{EF}f({\sim}(\mathrm{P}_{\leq x}p \wedge {\sim}\mathrm{P}_{>x}q{\rightarrow}\mathrm{EF}r))\\
&= \mathrm{EF}(f(\mathrm{P}_{\leq x}p \wedge {\sim}\mathrm{P}_{>x}q){\rightarrow}f({\sim}\mathrm{EF}r))\\
&= \mathrm{EF}(f(\mathrm{P}_{\leq x}p) \wedge f({\sim}\mathrm{P}_{>x}q){\rightarrow}\mathrm{AG}f({\sim}r))\\
&= \mathrm{EF}(\mathrm{P}_{\leq x}f(p) \wedge \mathrm{P}_{\leq x}f({\sim}q){\rightarrow}\mathrm{AG}r')\\
&= \mathrm{EF}(\mathrm{P}_{\leq x}p \wedge \mathrm{P}_{\leq x}q'{\rightarrow}\mathrm{AG}r')
\end{aligned}
$$

where p', q' are atomic formulas in pCTL. Thus, the formula ${\sim}\mathrm{AG}(\mathrm{P}_{\leq x}p \wedge {\sim}\mathrm{P}_{>x}q \rightarrow \mathrm{EF}r)$ in PpCTL is translated into the formula $\mathrm{EF}(\mathrm{P}_{\leq x}p \wedge \mathrm{P}_{\leq x}q'{\rightarrow}\mathrm{AG}r')$ in pCTL.

5. Bisimulations

In this section, we present the bisimulation theorem for PpCTL. Since concrete system models tend to be very large, the *state explosion problem* arises, and *abstraction techniques* are needed to reduce a large concrete model to a small abstract one. As presented in [4, 24], the following *bisimulation theorem* for CTL, which is useful for abstraction in model checking, is well-known:

> If two Kripke structures M and M' are bisimulation equivalent, then for every CTL-formula α, M satisfies α if and only if M' satisfies α.

This theorem guarantees that we can use an efficient small abstract structure that is bisimular to the given concrete large structure. The logic CTL and the bisimulation result addressed above are not sufficient for accommodating inconsistency-tolerant reasoning. Thus, the aim of this section is to extend the bisimulation framework of CTL to that of PpCTL.

Definition 5.1: Let $M = \langle S, S_0, R, \mu_s, L^+, L^- \rangle$ and $M' = \langle S', S'_0, R', \mu'_{s'}, L'^+, L'^- \rangle$ be ppk-structures with the same nonempty set AT $\subseteq$ ATOM.

A relation $B \subseteq S \times S'$ is a *bisimulation relation* between M and M' if and only if for all s and s', if $B(s,s')$ then the following conditions hold:

1. $L^+(s) = L'^+(s')$,
2. $L^-(s) = L'^-(s')$,
3. $\mu_s = \mu'_{s'}$
4. $\forall s_1[R(s,s_1)$ implies $\exists s'_1[R'(s',s'_1)$ and $B(s_1,s'_1)]]$.
5. $\forall s'_1[R'(s',s'_1)$ implies $\exists s_1[R(s,s_1)$ and $B(s_1,s'_1)]]$.

The structures M and M' are *bisimulation equivalent* if there exists a bisimulation relation B such that

1. $\forall s_0 \in S_0\ \exists s'_0 \in S'_0\ [B(s_0,s'_0)]$,
2. $\forall s'_0 \in S'_0\ \exists s_0 \in S_0\ [B(s_0,s'_0)]$.

Definition 5.2: Two paths $\pi = s_0, s_1, s_2, \ldots$ in a ppk-structure M and $\pi' = s'_0, s'_1, s'_2, \ldots$ in a ppk-structure M' are called *corresponding paths* if $\forall i \geq 0\ [B(s_i, s'_i)]$.

Lemma 5.3: Let s and s' be two states such that $B(s,s')$. Then for every path starting from s there is a corresponding path starting from s', and for every path starting from s' there is a corresponding path starting from s.

Proof. See [4]. ■

Lemma 5.4: Let M and M' be ppk-structures with the same nonempty set AT $\subseteq$ ATOM, and B be a bisimulation relation between M and M'. Let α be a PpCTL formula. Assume that $B(s,s')$ and that π in M and π' in M' are corresponding paths.

Then, for any formula α, we have:

1. $M, s \models^+ \alpha$ iff $M', s' \models^+ \alpha$,
2. $M, s \models^- \alpha$ iff $M', s' \models^- \alpha$.

Proof. This lemma is proved by (simultaneous) induction on the complexity of α. In the following, the ppk-structures M and M' are omitted from the expressions, since the structures are clear from the context.

- Base step:

  $\alpha \equiv p$ for $p \in$ AT. By the assumption $B(s,s')$, we have $L^+(s) = L'^+(s')$ and $L^-(s) = L'^-(s')$, and hence obtain (1) $s \models^+ p$ iff $s' \models^+ p$ and (2) $s \models^- p$ iff $s' \models^- p$.

- Induction step:

We show some cases.

1. Case ($\alpha \equiv \alpha_1 \vee \alpha_2$): (1) $s \models^+ \alpha_1 \vee \alpha_2$ iff $s \models^+ \alpha_1$ or $s \models^+ \alpha_2$ iff $s' \models^+ \alpha_1$ or $s' \models^+ \alpha_2$ (by induction hypothesis) iff $s' \models^+ \alpha_1 \vee \alpha_2$. (2) $s \models^- \alpha_1 \vee \alpha_2$ iff $s \models^- \alpha_1$ and $s \models^- \alpha_2$ iff $s' \models^- \alpha_1$ and $s' \models^- \alpha_2$ (by induction hypothesis) iff $s' \models^- \alpha_1 \vee \alpha_2$.
2. Case ($\alpha \equiv \sim\alpha_1$): (1) $s \models^+ \sim\alpha_1$ iff $s \models^- \alpha_1$ iff $s' \models^- \alpha_1$ (by induction hypothesis) iff $s' \models^+ \sim\alpha_1$. (2) $s \models^- \sim\alpha_1$ iff $s \models^+ \alpha_1$ iff $s' \models^+ \alpha_1$ (by induction hypothesis) iff $s' \models^- \sim\alpha_1$.
3. Case ($\alpha \equiv \neg\alpha_1$): (1) $s \models^+ \neg\alpha_1$ iff $s \not\models^+ \alpha_1$ iff $s' \not\models^+ \alpha_1$ (by induction hypothesis) iff $s' \models^+ \neg\alpha_1$. (2) $s \models^- \neg\alpha_1$ iff $s \models^+ \alpha_1$ iff $s' \models^+ \alpha_1$ (by induction hypothesis) iff $s' \models^- \neg\alpha_1$.
4. Case ($\alpha \equiv \mathrm{P}_{\leq x}\alpha_1$): (1) $s \models^+ \mathrm{P}_{\leq x}\alpha_1$ iff $\mu_s(\{\omega \in \Omega_s \mid \omega \models^+ \alpha_1\}) \leq x$ iff $\mu'_{s'}(\{\omega' \in \Omega_{s'} \mid \omega' \models^+ \alpha_1\}) \leq x$ (by hypothesis and induction hypothesis) iff $s' \models^+ \mathrm{P}_{\leq x}\alpha_1$. (2) $s \models^- \mathrm{P}_{\leq x}\alpha_1$ iff $\mu_s(\{\omega \in \Omega_s \mid \omega \models^- \alpha_1\}) > x$ iff $\mu'_{s'}(\{\omega' \in \Omega_{s'} \mid \omega' \models^- \alpha_1\}) > x$ (by hypothesis and induction hypothesis) iff $s' \models^- \mathrm{P}_{\leq x}\alpha_1$.
5. Case ($\alpha \equiv \mathrm{AX}\alpha_1$): (1) $s \models^+ \mathrm{AX}\alpha_1$ iff $\forall s_1 \in S\ [(s,s_1) \in R$ implies $s_1 \models^+ \alpha_1]$ iff $\forall s'_1 \in S'\ [(s',s'_1) \in R'$ implies $s'_1 \models^+ \alpha_1]$ (by hypothesis and induction hypothesis) iff $s' \models^+ \mathrm{AX}\alpha_1$. (2) $s \models^- \mathrm{AX}\alpha_1$ iff $\exists s_1 \in S\ [(s,s_1) \in R$ and $s_1 \models^+ \alpha_1]$ iff $\exists s'_1 \in S'\ [(s',s'_1) \in R'$ and $s'_1 \models^+ \alpha_1]$ (by hypothesis and induction hypothesis) iff $s' \models^- \mathrm{AX}\alpha_1$.
6. Case ($\alpha \equiv \mathrm{A}(\alpha_1 \mathrm{U} \alpha_2)$):

   (1) We only show the direction $s \models^+ \mathrm{A}(\alpha_1 \mathrm{U} \alpha_2)$ implies $s' \models^+ \mathrm{A}(\alpha_1 \mathrm{U} \alpha_2)$. The converse direction can be shown in a similar way. Suppose $s \models^+ \mathrm{A}(\alpha_1 \mathrm{U} \alpha_2)$. Then, we have: for all path $\pi \equiv s_0, s_1, s_2, \ldots$ where $s \equiv s_0$, there is a state s_k along π such that $[(s_k \models^+ \alpha_2)$ and $\forall j\ (0 \leq j < k$ implies $s_j \models^+ \alpha_2)]$. By Lemma 5.4, we have (*): there exists a corresponding path π' starting from s'_0, i.e., π and π' are corresponding paths. Then, by (*) and induction hypothesis, we have: for all path $\pi' \equiv s'_0, s'_1, s'_2, \ldots$ where $s' \equiv s'_0$, there is a state s'_k along π' such that $[(s'_k \models^+ \alpha_2)$ and $\forall j\ (0 \leq j < k$ implies $s'_j \models^+ \alpha_2)]$. This means $s' \models^+ \mathrm{A}(\alpha_1 \mathrm{U} \alpha_2)$.

   (2) We only show the direction $s \models^- \mathrm{A}(\alpha_1 \mathrm{U} \alpha_2)$ implies $s' \models^- \mathrm{A}(\alpha_1 \mathrm{U} \alpha_2)$. The converse direction can be shown in a similar way. Suppose $s \models^- \mathrm{A}(\alpha_1 \mathrm{U} \alpha_2)$. Then, we have: there is a path $\pi \equiv s_0, s_1, s_2, \ldots$ where $s \equiv s_0$, and for all states s_j along π, we have $[\forall i < j$ not-$(s_i \models^- \alpha_1)$ implies $s_j \models^- \alpha_2]$. By Lemma 5.4, we have (**): there exists a corresponding path π' starting from s'_0, i.e., π and π' are corresponding paths. Then, by (**) and induction hypothesis, we have: there is a path $\pi' \equiv s'_0, s'_1, s'_2, \ldots$ where $s' \equiv s'_0$, and for all states s'_j along π', we have $[\forall i < j$ not-$(s'_i \models^- \alpha_1)$ implies $s'_j \models^- \alpha_2]$. This means $s' \models^- \mathrm{A}(\alpha_1 \mathrm{U} \alpha_2)$.

■

The following theorem is a consequence of the preceding lemma.

Theorem 5.5: Let M and M' be ppk-structures and B be a bisimulation relation between M and M'. If $B(s,s')$, then for any formula α,

1. $M, s \models^+ \alpha$ iff $M', s' \models^+ \alpha$,
2. $M, s \models^- \alpha$ iff $M', s' \models^- \alpha$.

For any $* \in \{+,-\}$, $M \models^* \alpha$ is defined by $\forall s \in S[M,s \models^* \alpha]$. We then obtain the following bisimulation theorem.

Theorem 5.6—Bisimulation: Suppose that ppk-structures M and M' are bisimulation equivalent. For any formula α,

1. $M \models^+ \alpha$ iff $M' \models^+ \alpha$,
2. $M \models^- \alpha$ iff $M' \models^- \alpha$.

6. Illustrative Examples

6.1. SQL Injection Attack Detection Algorithm

SQL injection [25, 26] is one of the numerous malicious attack methods used to exploit security vulnerabilities on SQL database servers. An attacker sends injection codes through a network to illegally obtain stored information from the SQL database servers. According to an introductory article [26], methods intended as defense against SQL injection can be classified into three types: defensive coding, SQL injection vulnerabilities) detection, and SQL injection attack runtime prevention. Studies by various researchers have led to the proposal of a new approach to the latter of these three types of methods [9, 27, 28]. They utilized the contained rate of suspicious characters over the length of an input string. Consider an attempt by an automatic detection program to determine if the ith input string l_i $(i = 1,2,\ldots)$ to an SQL database server is obtained as a result of an SQL injection attack. Then, the *contained rate* p_i can be defined as:

$$p_i = \frac{\#S}{|l_i|}, \quad \ldots\ldots\ldots\ldots\ldots\ldots \quad (1)$$

where $\#S$ is the number of suspicious characters and $|l_i|$ is the length of the ith input string. Automatic detection with p_i is executed on the basis of the following rule:

$$h(p_i) = \begin{cases} 1 & \text{if} \quad p_i > \alpha; \\ 0 & \text{otherwise}, \end{cases} \quad \ldots\ldots\ldots \quad (2)$$

where $h(p_i) = 1$ indicates that the detected result is an attack string, $h(p_i) = 0$ implies that it is a normal string, and α is a predetermined *threshold value*. A set S contains some suspicious characters (e.g., a space, semi-colon, single quotation, etc.) in the input string of some SQL injection attacks. A learning algorithm of threshold value α from the observed data set of both input strings and their labels (attacks or normals) is proposed [28]. Furthermore, the detecting performance of discriminant functions $h(p_i)$ is considered [27].

Example 6.1: Suppose the unknown input string l_1 as "`DROP sampletable;--`" is sent to the SQL server. Let the elements of S be a space, semi-colon, and right parenthesis, and let the threshold value α be 0.08. Then, this input l_1 is detected as an attack string because the length $|l_1|$ is 19 and the suspicious characters contained in S is 2, the contained rate $p_1 = 2/19 = 0.105$, and hence p_1 is greater than α.

In an experiment [9], each *attack detection rate* μ_A and *normal detection rate* μ_N for the underlying characters were calculated by changing the threshold α. The *overall detection rate* μ is defined as the weighted average of μ_A and μ_N:

$$\mu = (1-\beta)\mu_A + \beta\mu_N, \quad \ldots\ldots\ldots\ldots \quad (3)$$

where a real number β, which satisfies $0 \leq \beta \leq 1$, is the weight of the normal string over the input strings. The use of the SQL injection attack detection algorithm explained above is assumed in the following discussion.

6.2. Representing Paraconsistency

Now, we consider some example formulas for SQL injection attacks. The paraconsistent negation connective $\sim\alpha$ in PpCTL is used to represent the negation of an uncertain or ambiguous concept "attack". If we cannot determine whether an input string is obtained by an SQL injection attack, then this concept is regarded as uncertain. The uncertain concept *attack* can be represented by asserting the inconsistent formula of the form:

$$attack \wedge \sim attack$$

where $\sim attack$ represents the uncertain negation information that can be true at the same time as *attack*, which represents positive information. This is well formalized because the formula of the form:

$$(attack \wedge \sim attack) \rightarrow \bot$$

is not valid in PpCTL.

We can also present the following formula:

$$\mathrm{EF}(attack \wedge \sim attack)$$

which implies:

> "There exists a situation in which a string input is considered to be obtained as both an SQL injection attack and a non-SQL injection attack, i.e., we cannot use the algorithm to determine whether a string was obtained from an attack."

In addition, we can present the following formula:

$$\mathrm{EF}(crashed \wedge \mathrm{AG}\ crashed)$$

which implies:

> "There is a situation in which a crashed database caused by an SQL injection attack will not function again."

6.3. Representing Probability

We can express Example 6.1 as the following formula:

$$\mathrm{AG}(\mathrm{P}_{\leq 0.08}\alpha \wedge (p_i < \alpha) \rightarrow \sim \mathrm{attack})$$

which implies:

> "If the threshold value α is at the most 8 percent and the contained rate p_i is greater than α, then the string was *probably not* obtained by an SQL injection attack, i.e., it can be regarded as a normal string."

Let μ_A and μ_N be an attack detection rate and a normal detection rate, respectively. Then, we can present the following formula:

$$\mathrm{AG}(\mathrm{P}_{\geq 0.08}\mu_A \wedge \mathrm{P}_{\geq 0.02}\mu_N \rightarrow \mathrm{attack})$$

which implies:

> "If the attack detection rate μ_A and the normal detection rate μ_N with respect to some fixed characters in the underlying string are at least 8 percent and at least 2 percent, respectively, then the string is obtained by an SQL injection attack, i.e., it is regarded as a malicious attack string."

Similarly, we can present the following formula:

$$\mathrm{AG}(\mathrm{P}_{<0.08}\mu_A \wedge \mathrm{P}_{<0.02}\mu_N \rightarrow \sim\mathrm{attack})$$

which implies:

> "The string entered by someone is *probably not* obtained by an SQL injection attack."

In addition, we present the following formula with the classical negation connective $\neg$:

$$\mathrm{AG}(\mathrm{P}_{<0.02}\mu_A \wedge \mathrm{P}_{<0.01}\mu_N \rightarrow \neg\mathrm{attack})$$

which implies:

> "The string entered by someone is *clearly not* obtained by an SQL injection attack, i.e., it is just a normal string."

6.4. Representing Some Experimental Facts

The single quotation mark "`'`" forms a set with the previous single quotation. A pair of single quotation marks appears, for instance, as "`uid='user01'`" which implies: "the user ID is user01." We can present this situation as the following formula:

$$\mathrm{AG}(\mathrm{singleQuotation} \wedge \mathrm{EF}\ \mathrm{singleQuotation} \rightarrow \sim\mathrm{attack})$$

which implies:

> "At any time, if a single quotation "`'`" appears in the string described in a web form, and the corresponding (closed) single quotation "`'`" eventually appears in the same string, then such an input string is probably not obtained as an SQL injection attack."

The statement "`OR 1=1`" is sometimes used in an attack string. Then, we present this situation as the following formula:

$$\mathrm{AG}(\mathrm{EF}\ \mathrm{or1{=}1} \rightarrow \mathrm{attack})$$

which implies:

> "At any time, if the statement "`OR 1=1`" eventually appears, then such an input string was probably obtained as an SQL injection attack."

7. Remarks on Extensions

In this section, we remark that PpCTL can be extended with the addition of a location operator $[l]$ which represents the location of propositions. An extension LPpCTL (*locative* PpCTL) on PpCTL is obtained from PpCTL by adding $[l]$. A formula of the form $[l]\alpha$ in LPpCTL can be interpreted as "proposition α holds at location l." The location operator is formulated in a similar setting as in [29, 30], which is regarded as a refinement of the original setting by Kobayashi et al. [31]. In [31], such a location operator was introduced using a structural congruence relation in formalizing a *distributed concurrent linear logic programming language*. In [29, 30], the framework of this original operator was improved as a purely logical formulation without any structural congruence relation.

Assuming a space domain Loc and the operator $[l]$ with $l \in \mathrm{Loc}$, we can interpret the satisfaction relation $(s,l) \models^{+} \alpha$ of LPpCTL as "proposition α holds at time (or state) s and location l." Then, various properties and situations with space and time can be expressed using LPpCTL-formulas. For example, the following liveness property can be expressed: "If we input the login-password of host computer Comp3 at one of the mobile computers Comp1 and Comp2, then we will eventually be able to login to Comp3." This is expressed formally as:

$$\mathrm{AG}([comp1]password \vee [comp2]password \rightarrow \mathrm{EF}[comp3]login)$$

where the space domain Loc is $\{comp1, comp2, comp3\}$.

We introduce the definition of formulas in LPpCTL.

Definition 7.1: Let Loc be a finite nonempty set of *locations*. Formulas α are defined by the following grammar, assuming $p \in \mathrm{ATOM}$, $x \in [0,1]$ and $l \in \mathrm{Loc}$:

$$\alpha ::= p \mid \alpha \rightarrow \alpha \mid \alpha \wedge \alpha \mid \alpha \vee \alpha \mid \neg\alpha \mid \sim\alpha \mid \mathrm{P}_{\leq x}\alpha \mid \mathrm{P}_{\geq x}\alpha \mid \mathrm{P}_{<x}\alpha \mid \mathrm{P}_{>x}\alpha \mid \mathrm{AX}\alpha \mid \mathrm{EX}\alpha \mid \mathrm{AG}\alpha \mid \mathrm{EG}\alpha \mid \mathrm{AF}\alpha \mid \mathrm{EF}\alpha \mid \mathrm{A}(\alpha \mathrm{U} \alpha) \mid \mathrm{E}(\alpha \mathrm{U} \alpha) \mid \mathrm{A}(\alpha \mathrm{R} \alpha) \mid \mathrm{E}(\alpha \mathrm{R} \alpha) \mid [l]\alpha.$$

We are ready to introduce LPpCTL.

Definition 7.2: A *locative paraconsistent probabilistic Kripke structure* (*lppk-structure* for short) is a structure $\langle \mathrm{Loc}, S, S_0, R, \mu_s, L^{+}, L^{-} \rangle$ such that

1. Loc is a finite nonempty set of locations,
2. $\langle S, S_0, R, \mu_s, L^{+}, L^{-} \rangle$ is a ppk-structure.

A *path* in an lppk-structure is defined similarly as in a ppk-structure.

Definition 7.3—LPpCTL: Let AT be a nonempty subset of ATOM. *Satisfaction relations* $\models^{+}$ and $\models^{-}$ on an

lppk-structure $M = \langle \mathrm{Loc}, S, S_0, R, \mu_s, L^+, L^- \rangle$ are defined inductively as follows (s represents a state in S and l, k represent locations in Loc):

1. for any $p \in \mathrm{AT}$, $M,(s,l) \models^+ p$ iff $p \in L^+(s)$ and $l \in \mathrm{Loc}$,
2. $M,(s,l) \models^+ [k]\alpha$ iff $M,(s,k) \models^+ \alpha$,
3. $M,(s,l) \models^+ \alpha_1 \to \alpha_2$ iff $M,(s,l) \models^+ \alpha_1$ implies $M,(s,l) \models^+ \alpha_2$,
4. $M,(s,l) \models^+ \alpha_1 \wedge \alpha_2$ iff $M,(s,l) \models^+ \alpha_1$ and $M,(s,l) \models^+ \alpha_2$,
5. $M,(s,l) \models^+ \alpha_1 \vee \alpha_2$ iff $M,(s,l) \models^+ \alpha_1$ or $M,(s,l) \models^+ \alpha_2$,
6. $M,(s,l) \models^+ \neg\alpha_1$ iff $M,(s,l) \not\models^+ \alpha_1$,
7. $M,(s,l) \models^+ {\sim}\alpha$ iff $M,(s,l) \models^- \alpha$,
8. for any $x \in [0,1]$, $M,(s,l) \models^+ \mathrm{P}_{\leq x}\alpha$ iff $\mu_s(\{w \in \Omega_s \mid M,(s,l) \models^+ \alpha\}) \leq x$,
9. for any $x \in [0,1]$, $M,(s,l) \models^+ \mathrm{P}_{\geq x}\alpha$ iff $\mu_s(\{w \in \Omega_s \mid M,(s,l) \models^+ \alpha\}) \geq x$,
10. for any $x \in [0,1]$, $M,(s,l) \models^+ \mathrm{P}_{< x}\alpha$ iff $\mu_s(\{w \in \Omega_s \mid M,(s,l) \models^+ \alpha\}) < x$,
11. for any $x \in [0,1]$, $M,(s,l) \models^+ \mathrm{P}_{> x}\alpha$ iff $\mu_s(\{w \in \Omega_s \mid M,(s,l) \models^+ \alpha\}) > x$,
12. $M,(s,l) \models^+ \mathrm{AX}\alpha$ iff $\forall s_1 \in S$ $[(s,s_1) \in R$ implies $M,(s_1,l) \models^+ \alpha]$,
13. $M,(s,l) \models^+ \mathrm{EX}\alpha$ iff $\exists s_1 \in S$ $[(s,s_1) \in R$ and $M,(s_1,l) \models^+ \alpha]$,
14. $M,(s,l) \models^+ \mathrm{AG}\alpha$ iff for all paths $\pi \equiv s_0,s_1,s_2,...$, where $s \equiv s_0$, and all states s_i along π, we have $M,(s_i,l) \models^+ \alpha$,
15. $M,(s,l) \models^+ \mathrm{EG}\alpha$ iff there is a path $\pi \equiv s_0,s_1,s_2,...$, where $s \equiv s_0$, and for all states s_i along π, we have $M,(s_i,l) \models^+ \alpha$,
16. $M,(s,l) \models^+ \mathrm{AF}\alpha$ iff for all paths $\pi \equiv s_0,s_1,s_2,...$, where $s \equiv s_0$, there is a state s_i along π such that $M,(s_i,l) \models^+ \alpha$,
17. $M,(s,l) \models^+ \mathrm{EF}\alpha$ iff there is a path $\pi \equiv s_0,s_1,s_2,...$, where $s \equiv s_0$, and for some state s_i along π, we have $M,(s_i,l) \models^+ \alpha$,
18. $M,(s,l) \models^+ \mathrm{A}(\alpha_1 \mathrm{U} \alpha_2)$ iff for all paths $\pi \equiv s_0,s_1,s_2,...$, where $s \equiv s_0$, there is a state s_k along π such that $[(M,(s_k,l) \models^+ \alpha_2)$ and $\forall j\ (0 \leq j < k$ implies $M,(s_j,l) \models^+ \alpha_1)]$,
19. $M,(s,l) \models^+ \mathrm{E}(\alpha_1 \mathrm{U} \alpha_2)$ iff there is a path $\pi \equiv s_0,s_1,s_2,...$, where $s \equiv s_0$, and for some state s_k along π, we have $[(M,(s_k,l) \models^+ \alpha_2)$ and $\forall j\ (0 \leq j < k$ implies $M,(s_j,l) \models^+ \alpha_1)]$,
20. $M,(s,l) \models^+ \mathrm{A}(\alpha_1 \mathrm{R} \alpha_2)$ iff for all paths $\pi \equiv s_0,s_1,s_2,...$, where $s \equiv s_0$, and all states s_j along π, we have $[\forall i < j$ not-$[M,(s_i,l) \models^+ \alpha_1]$ implies $M,(s_j,l) \models^+ \alpha_2]$,
21. $M,(s,l) \models^+ \mathrm{E}(\alpha_1 \mathrm{R} \alpha_2)$ iff there is a path $\pi \equiv s_0,s_1,s_2,...$, where $s \equiv s_0$, and for all states s_j along π, we have $[\forall i < j$ not-$[M,(s_i,l) \models^+ \alpha_1]$ implies $M,(s_j,l) \models^+ \alpha_2]$,
22. for any $p \in \mathrm{AT}$, $M,(s,l) \models^- p$ iff $p \in L^-(s)$ and $l \in \mathrm{Loc}$,
23. $M,(s,l) \models^- [k]\alpha$ iff $M,(s,k) \models^- \alpha$,
24. $M,(s,l) \models^- \alpha_1 \to \alpha_2$ iff $M,(s,l) \models^+ \alpha_1$ and $M,(s,l) \models^- \alpha_2$,
25. $M,(s,l) \models^- \alpha_1 \wedge \alpha_2$ iff $M,(s,l) \models^- \alpha_1$ or $M,(s,l) \models^- \alpha_2$,
26. $M,(s,l) \models^- \alpha_1 \vee \alpha_2$ iff $M,(s,l) \models^- \alpha_1$ and $M,(s,l) \models^- \alpha_2$,
27. $M,(s,l) \models^- \neg\alpha_1$ iff $M,(s,l) \models^+ \alpha_1$,
28. $M,(s,l) \models^- {\sim}\alpha_1$ iff $M,(s,l) \models^+ \alpha_1$,
29. for any $x \in [0,1]$, $M,(s,l) \models^- \mathrm{P}_{\leq x}\alpha$ iff $\mu_s(\{w \in \Omega_s \mid M,(s,l) \models^- \alpha\}) > x$,
30. for any $x \in [0,1]$, $M,(s,l) \models^- \mathrm{P}_{\geq x}\alpha$ iff $\mu_s(\{w \in \Omega_s \mid M,(s,l) \models^- \alpha\}) < x$,
31. for any $x \in [0,1]$, $M,(s,l) \models^- \mathrm{P}_{< x}\alpha$ iff $\mu_s(\{w \in \Omega_s \mid M,(s,l) \models^- \alpha\}) \geq x$,
32. for any $x \in [0,1]$, $M,(s,l) \models^- \mathrm{P}_{> x}\alpha$ iff $\mu_s(\{w \in \Omega_s \mid M,(s,l) \models^- \alpha\}) \leq x$,
33. $M,(s,l) \models^- \mathrm{AX}\alpha$ iff $\exists s_1 \in S$ $[(s,s_1) \in R$ and $M,(s_1,l) \models^- \alpha]$,
34. $M,(s,l) \models^- \mathrm{EX}\alpha$ iff $\forall s_1 \in S$ $[(s,s_1) \in R$ implies $M,(s_1.l) \models^- \alpha]$,
35. $M,(s,l) \models^- \mathrm{AG}\alpha$ iff there is a path $\pi \equiv s_0,s_1,s_2,...$, where $s \equiv s_0$, and for some state s_i along π, we have $M,(s_i,l) \models^- \alpha$,
36. $M,(s,l) \models^- \mathrm{EG}\alpha$ iff for all paths $\pi \equiv s_0,s_1,s_2,...$, where $s \equiv s_0$, there is a state s_i along π such that $M,(s_i,l) \models^- \alpha$,
37. $M,(s,l) \models^- \mathrm{AF}\alpha$ iff there is a path $\pi \equiv s_0,s_1,s_2,...$, where $s \equiv s_0$, and for all states s_i along π, we have $M,(s_i,l) \models^- \alpha$,
38. $M,(s,l) \models^- \mathrm{EF}\alpha$ iff for all paths $\pi \equiv s_0,s_1,s_2,...$, where $s \equiv s_0$, and all states s_i along π, we have $M,(s_i,l) \models^- \alpha$,
39. $M,(s,l) \models^- \mathrm{A}(\alpha_1 \mathrm{U} \alpha_2)$ iff there is a path $\pi \equiv s_0,s_1,s_2,...$, where $s \equiv s_0$, and for all states s_j along π, we have $[\forall i < j$ not-$[M,(s_i,l) \models^- \alpha_1]$ implies $M,(s_j,l) \models^- \alpha_2]$,
40. $M,(s,l) \models^- \mathrm{E}(\alpha_1 \mathrm{U} \alpha_2)$ iff for all paths $\pi \equiv s_0,s_1,s_2,...$, where $s \equiv s_0$, and for all states s_j along π, we have $[\forall i < j$ not-$[M,(s_i,l) \models^- \alpha_1]$ implies $M,(s_j,l) \models^- \alpha_2]$,
41. $M,(s,l) \models^- \mathrm{A}(\alpha_1 \mathrm{R} \alpha_2)$ iff there is a path $\pi \equiv s_0,s_1,s_2,...$, where $s \equiv s_0$, and for some state s_k along π, we have $[(M,(s_k,l) \models^- \alpha_2)$ and $\forall j\ (0 \leq j < k$ implies $M,(s_j,l) \models^- \alpha_1)]$,
42. $M,(s,l) \models^- \mathrm{E}(\alpha_1 \mathrm{R} \alpha_2)$ iff for all paths $\pi \equiv s_0,s_1,s_2,...$, where $s \equiv s_0$, there is a state s_k along π such that $[(M,(s_k,l) \models^- \alpha_2)$ and $\forall j\ (0 \leq j < k$ implies $M,(s_j,l) \models^- \alpha_1)]$.

Definition 7.4: A formula α is *valid* (satisfiable) in LPpCTL if $M,(s,l) \models^+ \alpha$ holds for any (some) lppk-structure $M = \langle \mathrm{Loc}, S, S_0, R, \mu_s, L^+, L^- \rangle$, any (some) $s \in S$, any (some) $l \in \mathrm{Loc}$ and any (some) satisfaction relations $\models^+$ and $\models^-$ on M.

Definition 7.5: Let M be an lppk-structure $\langle \mathrm{Loc}, S, S_0, R, \mu_s, L^+, L^- \rangle$ for LPpCTL, and $\models^+$ and $\models^-$ be satisfaction relations on M. Then, the *positive and negative model checking problems* for LPpCTL are respectively defined by: for any formula α, find the sets $\{s \in S \mid M,(s,l) \models^+ \alpha\}$ and $\{s \in S \mid M,(s,l) \models^- \alpha\}$.

The proposed setting of the location operator $[l]$ represents the discrete space interpretation in which a location is regarded as a point and is independent of other locations. To introduce $[l]$ is to allow the following axiom schemes with respect to $[l]$: For any $l, l_i, l_j \in \mathrm{Loc}$,

1. $[l_i][l_j]\alpha \leftrightarrow [l_i]\alpha$,
2. $[l](\alpha \sharp \beta) \leftrightarrow ([l]\alpha)\sharp([l]\beta)$ where $\sharp \in \{\wedge, \vee, \to\}$,
3. $[l](\dagger(\alpha \sharp \beta)) \leftrightarrow \dagger(([l]\alpha)\sharp([l]\beta))$ where $\dagger \in \{\mathrm{A}, \mathrm{E}\}$ and $\sharp \in \{\mathrm{U}, \mathrm{R}\}$,

4. $[l](\sharp\alpha) \leftrightarrow \sharp([l]\alpha)$ where $\sharp \in \{\sim, \neg, \mathrm{AX}, \mathrm{EX}, \mathrm{AG}, \mathrm{EG}, \mathrm{AF}, \mathrm{EF}, \mathrm{P}_{\leq x}, \mathrm{P}_{\geq x}, \mathrm{P}_{<x}, \mathrm{P}_{>x}\}$,

The first axiom scheme $[l_i][l_j]\alpha \leftrightarrow [l_i]\alpha$ displayed above intuitively means that each location l is the absolute address of locations, i.e., the location l refers to the same location anywhere [31]. The other axiom schemes displayed above intuitively mean that the truth is time- and space-independent, i.e., "space" is almost independent of "time". As mentioned in [29, 30], the following inference rule called *space induction rule* is also true:

If $(\forall l \in \mathrm{Loc})([l]\alpha)$, then α.

In this rule, if $[l_1]\alpha$, $[l_2]\alpha$ and $[l_3]\alpha$ with $\mathrm{Loc} = \{l_1, l_2, l_3\}$ hold, then α holds. It is also remarked that if $\mathrm{Loc} = \{l\}$, then $[l]$ derives the modal logic S4-like axiom schemes:

1. $[l](\alpha \rightarrow \beta) \rightarrow ([l]\alpha \rightarrow [l]\beta)$,
2. $[l]\alpha \rightarrow [l][l]\alpha$,
3. $[l]\alpha \rightarrow \alpha$.

Thus, the operator $[l]$ is more expressive (or stronger) than the S4-type modal operator. Since the case that Loc is empty corresponds to the PpCTL case, LPpCTL is regarded as a natural generalization and extension of PpCTL.

We have not yet obtained a decidability result for model checking based on LPpCTL. On the other hand, a bisimulation result for LPpCTL is obtained.

Definition 7.6: Let $M = \langle \mathrm{Loc},\ S, S_0, R, L^+, L^- \rangle$ and $M' = \langle \mathrm{Loc},\ S', S'_0, R', L'^+, L'^- \rangle$ be lppk-structures with the common (nonempty) sets AT ($\subseteq$ ATOM) and Loc. The definition of bisimulation w.r.t. M and M' is almost the same as that in Definition 5.1, since the space domain Loc is independent of this definition: Of course, the conditions $L^+(s) = L'^+(s')$ and $L^-(s) = L'^-(s')$ in Definition 5.1 must be replaced by $L^+(s,l) = L'^+(s',l)$ and $L^-(s,l) = L'^-(s',l)$, respectively. The notion of the "corresponding paths" in lppk-structures is the same as that in Definition 5.2,

The lemma concerning the notion of the "corresponding paths" also holds for the same setting as Lemma 5.3.

Lemma 7.7: Let M and M' be lppk-structures with the same nonempty set AT $\subseteq$ ATOM, and B be a bisimulation relation between M and M'. Let α be a LPpCTL formula. Assume that $B(s,s')$ and that π in M and π' in M' are corresponding paths.

Then, for any formula α, we have:

1. $M, (s,l) \models^+ \alpha$ iff $M', (s',l) \models^+ \alpha$,
2. $M, (s,l) \models^- \alpha$ iff $M', (s',l) \models^- \alpha$.

Proof. This lemma is proved by (simultaneous) induction on the complexity of α. In the following, the ppk-structures M and M' are omitted from the expressions, since the structures are clear from the context. We only show the following case. The other cases are similar to the cases of PpCTL.

Case $(\alpha \equiv [k]\alpha_1)$: Let $* \in \{+, -\}$. $(s,l) \models^* [k]\alpha_1$ iff $(s,k) \models^* \alpha_1$ iff $(s',k) \models^* \alpha_1$ (by induction hypothesis) iff $(s',l) \models^* [k]\alpha_1$. ■

We now define $M \models^* \alpha$ as $\forall s \in S\, \forall l \in \mathrm{Loc}\, [M, (s,l) \models^* \alpha]$. Then, $M \models^* \alpha$ is intuitively interpreted as follows: "If a proposition α can be verified (or refuted) at any time in the future for all spaces in a world M, then the proposition is the eternal truth (or falsehood) in the world".

We then obtain the following bisimulation theorem.

Theorem 7.8—Bisimulation: Suppose that lppk-structures M and M' are bisimulation equivalent. For any formula α,

1. $M \models^+ \alpha$ iff $M' \models^+ \alpha$,
2. $M \models^- \alpha$ iff $M' \models^- \alpha$.

8. Conclusions and Related Works

In this paper, paraconsistent probabilistic computation tree logic (PpCTL) was introduced and studied. PpCTL was constructed by combining two existing extended temporal logics: Paraconsistent computation tree logic (PCTL) and probabilistic computation tree logic (pCTL). Then, a theorem for embedding PpCTL into pCTL was proven using translation, which is independent of the probability measure setting. A relative decidability theorem for PpCTL, which states that the decidability of pCTL implies that of PpCTL, was also obtained as a corollary of this embedding theorem. This relative decidability theorem indicates that we can reuse some existing pCTL-based verification algorithms. Some illustrative examples for describing an SQL injection attack detection algorithm, involving the use of PpCTL, were also presented to highlight the virtues of combining paraconsistency (in PCTL) and probability (in pCTL).

Some remarks are given as follows. A translation from PpCTL into PCTL was not provided in this paper, although a translation from PpCTL into pCTL was given. The issue for obtaining a translation from PpCTL (pCTL) into PCTL (CTL, resp.) has not been solved yet, because a formula with probabilistic operators that have probability measures is difficult to translate into a non-probabilistic formula of PCTL or CTL. In the meantime, we would like to extend the proposed embedding-based method to obtain an extended PpCTL with the *sequence modal operator* which was introduced for expressing ontological or hierarchical information (see e.g, [32–37]). This issue would need to be addressed in future. We would also like to show that the proposed embedding-based method for extending CTL with paraconsistency and probability is applicable to other temporal logics such as linear-time temporal logic (LTL) and full computation-tree logic (CTL*). Namely, we would like to introduce PpLTL and PpCTL* in a similar manner to these other logics, and to prove their corresponding embedding and relative decidability theorems. This issue would also need to be resolved in future.

The remainder of this paper addresses some related

works. Although the idea of combining paraconsistency and probability within a temporal logic is new, the idea of introducing a paraconsistent computation tree logic is not. In this study, PCTL [6, 7] was used as a base logic for constructing PpCTL. However, there are some other paraconsistent variants of CTL. For example, a *multi-valued computation tree logic*, χCTL, was introduced by Easterbrook and Chechik [38], and a *quasi-classical temporal logic*, QCTL, was proposed by Chen and Wu [1]. PCTL was introduced as an alternative to these logics. In addition, an extension PCTL* of PCTL has also been studied from the viewpoint of bisimulations for paraconsistent Kripke structures in paraconsistent model checking [39]. Another extension of PCTL was also studied in [32] for verifying student learning processes in learning support systems.

Compared with paraconsistent CTLs, several studies have been reported on probabilistic temporal logics, including probabilistic CTLs. The study in [40] is a typical example of such a study. In [40], a probabilistic and real-time extension of CTL, also called PCTL, was introduced and investigated on the basis of an interpretation of *discrete time Markov chains*. In contrast to the probabilistic frameworks of pCTL and PpCTL, the notion of probability in PCTL is assigned to all the temporal operators in PCTL. For example, a PCTL formula with the form $\mathrm{G}^{\leq t}_{\geq p}\alpha$ implies "the formula α holds continuously for t time units with a probability of at least p."

Some works on non-classical logics with the sequence modal operator are explained as follows, although these logics have no probability account. The sequence modal operator can be used for a wide range of non-classical logics. For example, an extended CTL* with the sequence modal operator was studied in [35], wherein it was shown that the sequence modal operator is applicable to certain ontological descriptions. An extended LTL with the sequence modal operator was also studied in [36], wherein it was observed that the sequence modal operator is useful for specifying some time-dependent secure authentication systems. A study of an extension of the *logic BI of bunched implications*, which was obtained by adding the sequence modal operator, was reported [34], and it was shown that the completeness theorem with respect to an extension of the Grothendieck topological semantics for BI holds for such an extension of BI.

Acknowledgements

We would like to thank the anonymous referees for their valuable comments. This work was supported by JSPS KAKENHI Grant (C) JP26330263.

References:

[1] D. Chen and J. Wu, "Reasoning about inconsistent concurrent systems: A non-classical temporal logic," Proc. of the 32nd Conf. on Current Trends in Theory and Practice of Computer Science (SOFSEM 2006), Lecture Notes in Computer Science, Vol.3831, pp. 207-217, 2006.

[2] A. Bianco and L. de Alfaro, "Model checking of probabilistic and nondeterministic systems," Proc. of the 15th Conf. on Foundations of Software Technology and Theoretical Computer Science (FSTTCS 1995), Lecture Notes in Computer Science, Vol.1026, pp. 499-513, 1995.

[3] E. M. Clarke and E. A. Emerson, "Design and synthesis of synchronization skeletons using branching time temporal logic," Proc. of the Workshop on Logics of Programs, Lecture Notes in Computer Science, Vol.131, pp. 52-71, 1981.

[4] E.M. Clarke, O. Grumberg, and D.A. Peled, "Model checking," The MIT Press, 1999.

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

[6] N. Kamide and K. Kaneiwa, "Paraconsistent negation and classical negation in computation tree logic," Proc. of the 2nd Int. Conf. on Agents and Artificial Intelligence (ICAART 2010), Vol. 1, pp. 464-469, 2010.

[7] K. Kaneiwa and N. Kamide, "Paraconsistent computation tree logic," New Generation Computing, Vol.29, No.4, pp. 391-408, 2011.

[8] A. Aziz, V. Singhal, and F. Balarin, "It usually works: The temporal logic of stochastic systems," Proc. of the 7th Int. Conf. on Computer Aided Verification (CAV 1995), Lecture Notes in Computer Science 939, pp. 155-165, 1995.

[9] M. Sonoda, T. Matsuda, D. Koizumi, and S. Hirasawa, "On automatic detection of SQL injection attacks by the feature extraction of the single character," Proc. of the 4th Int. Conf. on Security of Information and Networks (SIN 2011), pp. 81-86, 2011.

[10] W. Carnielli, M. Coniglio, D.M. Gabbay, P. Gouveia, and C. Sernadas, "Analysis and synthesis of logics: How to cut and paste reasoning systems," Applied Logic Series, Vol.35, Springer, 2008.

[11] A. Almukdad and D. Nelson, "Constructible falsity and inexact predicates," J. of Symbolic Logic, Vol.49, pp. 231-233, 1984.

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

[13] G. Priest and R. Routley, Introduction: paraconsistent logics, Studia Logica, 43, pp. 3-16, 1982.

[14] N.C.A. da Costa, J. Béziau and O.A. Bueno, Aspects of paraconsistent logic, Bulletin of the IGPL 3 (4), pp. 597-614, 1995.

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

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

[17] N. Kamide, Inconsistency-tolerant bunched implications, Int. J. of Approximate Reasoning 54 (2), pp. 343-353, 2013.

[18] N. Kamide and H. Wansing, Combining linear-time temporal logic with constructiveness and paraconsistency, J. of Applied Logic 8 (1), pp. 33-61, 2010.

[19] N. Kamide and H. Wansing, A paraconsistent linear-time temporal logic, Fundamenta Informaticae 106 (1), pp. 1-23, 2011.

[20] Y. Gurevich, Intuitionistic logic with strong negation, Studia Logica 36, pp. 49-59, 1977.

[21] W. Rautenberg, Klassische und nicht-klassische Aussagenlogik, Vieweg, Braunschweig, 1979.

[22] N.N. Vorob'ev, A constructive propositional calculus with strong negation (in Russian), Doklady Akademii Nauk SSR 85, pp. 465-468, 1952.

[23] N. Kamide and D. Koizumi, Combining paraconsistency and probability in CTL, Proc. of the 7th Int. Conf. on Agents and Artificial Intelligence (ICAART 2015), pp. 285-293, 2015.

[24] M.C. Browne, E.M. Clarke, and O. Grumberg, Characterizing finite Kripke structures in propositional temporal logic, Theoretical Computer Science 59, pp. 115-131, 1988.

[25] J. Clarke, SQL injection attacks and defense, 2nd Edition, Syngress Publishing, 2009.

[26] Lwin Khin Shar and Hee Beng Kuan Tan, Defeating SQL Injection, Computer, Vol.46, Issue 3, IEEE Computer Society, pp. 69-77, 2013.

[27] T. Matsuda, D. Koizumi, M. Sonoda, and S. Hirasawa, On predictive errors of SQL injection attack detection by the feature of the single character, Proc. of the IEEE Int. Conf. on Systems, Man and Cybernetics, (SMC 2011), pp. 1722-1727, 2011.

[28] D. Koizumi, T. Matsuda, M. Sonoda, and S. Hirasawa, A learning algorithm of threshold value on the automatic detection of SQL injection attack, Proc. of the 2012 Int. Conf. on Parallel and Distributed Processing Techniques and Applications (PDPTA 2012), Vol.2, pp. 933-937, 2012.

[29] N. Kamide, A spatial modal logic with a location interpretation, Mathematical Logic Quarterly,Vol.51, No.4, pp. 331-341, 2005.

[30] N. Kamide, Linear and affine logics with temporal, spatial and epistemic operators, Theoretical Computer Science, Vol.353, No.1-3, pp. 165-207, 2006.

[31] N. Kobayashi, T. Shimizu, and A. Yonezawa, Distributed concurrent linear logic programming, Theoretical Computer Science, Vol.227, pp. 185-220, 1999.
[32] N. Kamide, Modeling and verifying inconsistency-tolerant temporal reasoning with hierarchical information: Dealing with students' learning processes, Proc. of the IEEE Int. Conf. on Systems, Man, and Cybernetics (SMC 2013), pp. 1859-1864, 2013.
[33] N. Kamide, Inconsistency-tolerant temporal reasoning with hierarchical information, Information Sciences, Vol.320, pp. 140-155, 2015.
[34] N. Kamide, Bunched sequential information, J. of Applied Logic, Vol.15, pp. 150-170, 2016.
[35] N. Kamide and K. Kaneiwa, Extended full computation-tree logic with sequence modal operator: Representing hierarchical tree structures, Proc. of the 22nd Australasian Joint Conf. on Artificial Intelligence (AI'09), Lecture Notes in Artificial Intelligence, Vol.5866, pp. 485-494, 2009.
[36] K. Kaneiwa and N. Kamide, Sequence-indexed linear-time temporal logic: Proof system and application, Applied Artificial Intelligence, Vol.24, pp. 896-913, 2010.
[37] K. Kaneiwa and N. Kamide, Conceptual modeling in full computation-tree logic with sequence modal operator, Int. J. of Intelligent Systems, Vol.26, No.7, pp. 636-651, 2011.
[38] S. Easterbrook, and M. Chechik, A framework for multi-valued reasoning over inconsistent viewpoints, Proc. of the 23rd Int. Conf. on Software Engineering (ICSE 2001), pp. 411-420, 2001.
[39] N. Kamide, Extended full computation tree logics for paraconsistent model checking, Logic and Logical Philosophy, Vol.15, No.3, pp. 251-276, 2006.
[40] H. Hansson and B. Jonsson, A logic for reasoning about time and reliability, Formal Aspects of Computing, Vol.6, No.5, pp. 512-535, 1994.

Name:
Norihiro Kamide

Affiliation:
Faculty of Science and Engineering, Department of Information and Electronic Engineering, Teikyo University

Address:
1-1 Toyosatodai, Utsunomiya, Tochigi 320-8551, Japan

Brief Biographical History:
2000 Received his Ph.D. in Information Science from Japan Advanced Institute of Science and Technology

Main Works:
- Mathematical logic
- Philosophical logic
- Computer science logic

Membership in Academic Societies:
- Japanese Society for Artificial Intelligence (JSAI)
- Japan Society for Software Science and Technology
- Mathematical Society of Japan

Name:
Daiki Koizumi

Affiliation:
Faculty of Commerce, Department of Information and Management Science, Otaru University of Commerce

Address:
3-5-21 Midori, Otaru, Hokkaido 047-8501, Japan

Brief Biographical History:
2010 Received his Ph.D. in Engineering from Waseda University

Main Works:
- Bayes decision theory, Bayesian statistics, and their applications
- Information security

Membership in Academic Societies:
- International Society for Bayesian Analysis
- The Institute of Electronics, Information and Communication Engineers (IEICE)