A 3–Valued Contraction Model Checking Game: Deciding on the World of Partial Information

Jandson S. Ribeiro* and Aline Andrade**

Distributed Systems Laboratory (LaSiD)
Computer Science Department – Mathematics Institute
Federal University of Bahia, Salvador, Bahia, Brazil
http://www.lasid.ufba.br
jandsonsan@dcc.ufba.br and aline@ufba.br

Abstract. In this work we address the problem of model checking a desired property specified in Computation Tree Logic (CTL) in the presence of partial information. The Kripke Modal Transition System (KMTS) is used for modelling due its capacity to represent indefinitions explicitly which enables a KMTS interpretation as a set of Kripke structures. In this interpretation a specific model checking algorithm is required that can return one of the three possible values: *true* when all Kripke models of the set satisfy the property, *false* when no Kripke models of the set satisfy the property and *indefinite* when some models satisfy and others do not. To the best of our knowledge the literature lacks a KMTS model checking algorithm that fits this interpretation and in this paper we present an algorithm based on a game approach called a Contraction Model Checking algorithm for this purpose.

Keywords: Kripke Modal Transition System (KMTS), Model Checking Game, Partial Information

1 Introduction

In this work, we address the problem of verifying whether a model of a system in the presence of partial information satisfies a required property. It is desirable to express partial information explicitly, mainly in the preliminary phases of system development, in order to better give support to the evolution of the model as new information is acquired.

Model checking tools use algorithms that in general are defined over Kripke structures that do not express partial information explicitly. However, 3-valued model checking has been proposed which returns an undetermined value besides True and False to work with modal transition systems which express indeterminations explicitly. An example of such a structure is the Kripke Modal Transition System - KMTS [8].

* This author is supported by the grant # 4576/2014, Bahia Research Foundation (FAPESB).

** This author is supported by the grant # 447178/2014-8, Brazilian Research Council (CNPq).

The Kripke Modal Transition System structures express indetermination in states and in transitions, which induce a set of possible Kripke models by transforming indeterminations into determinations. In this work we propose a KMTS model checking algorithm according to this interpretation. This KMTS model checking algorithm should consider three possible results when verifying a property: *true* (T) when all Kripke models of the set satisfy the property, *false* (F) when no Kripke models of the set satisfy the property and *indefinite* ($\perp$) when some models satisfy and others do not. In [5], Grumberg proposes a 3-valued model checking algorithm based on games to verify CTL properties over KMTS using Kleene three-valued logic [8,10]. The algorithm uses a colouring function over a game board for model checking. We take this algorithm as a reference to develop the model checking algorithm presented in this paper. However, the interpretation of a KTMS considered in [5] and [11] is different from ours. There a KMTS is interpreted as an abstraction of a concrete Kripke structure and the Kleene three valued logic is suitable for this. In interpreting a KMTS as a set of Kripke models, however, we cannot reason with this logic because the $\perp$ value will not be compositional over conjunctions and disjunctions, i.e., if φ is $\perp$ and ψ is $\perp$, $\varphi \wedge \psi$ is not necessarily $\perp$.

In order to achieve a model checking that fits our interpretation, we consider the truth value of a formula in the model checking process represented as a set of KMTSs. Over truth values (sets of KMTS models) of two formulas φ and ψ we define a contraction operation that can calculate the truth value of formulas composed of φ and ψ. A proper colouring function is defined over a game board using this contraction operation.

The initial motivation of this work is to support the framework for KMTS revision combined with a model checking game proposed in [6], where a KMTS is interpreted as a set of Kripke models. The algorithms for revision in [6] took as reference the Grumberg 3-valued model checking game and can only give partial results because of the difference in semantics reported above. Our Contraction Model Checking algorithm solves this problem enabling a KMTS complete revision. We forecast that this algorithm can also be used in other contexts and can be adjusted to other logics unlike CTL.

With regards to the organization of this paper, in Section 2 we present CTL, Kripke structures and KMTS interpreted as a set of CTL models. Section 3 presents the semantics of CTL w.r.t KMTS interpreted as a set of Kripke structures. In Section 4 we present KMTS set operations, including the contraction operation. We present in Section 5 our Contraction Model Checking and finally in Section 6 we present related works and make some final considerations.

2 Computation Tree Logic and Kripke Structures

Computation Tree Logic (CTL) [3,9] is a temporal logic that presupposes a branched representation of the future over sequences of states of a Kripke structure which forms a computation tree. Path quantifiers can be used to make reference to a future or all futures.

Definition 1. *A Kripke structure is a tuple* $K = (AP, S, S_0, R, L)$ *where* AP *is a set of atomic propositions;* S *is a finite set of states,* $S_0 \subseteq S$ *is the set of initial states,* $R \subseteq S \times S$ *is the transition relation over* S*, and* $L : S \to 2^{AP}$ *is a labelling function of truth assignment over states.*[1]

We denote a transition $(s, s') \in R$ of a Kripke structure K by $s \to s'$. Furthermore, to inform that $s \to s'$ belongs to the set transitions of K we write $s \to s' \in K$.

Definition 2. *Let* l *be a literal. A CTL formula* ϕ *in its negation normal formal is defined as follows:*

$$\phi ::= \top \mid F \mid l \mid (\phi \vee \phi) \mid (\phi \wedge \phi) \mid EX\phi \mid AX\phi \mid$$
$$E[\phi U \phi] \mid A[\phi U \phi] \mid E[\phi R \phi] \mid A[\phi R \phi]$$

In Definition 2, A and E are path operators meaning for all paths and exists a path, respectively. The operators X, U and R mean, respectively, next state, until (in the sense that the left ϕ must hold along the path until the right ϕ holds) and release (the until dual operator). Excluding the conjunction and the disjunction, all the other operators must be bound by path operators. The complete semantics of the CTL formulas can be found in [3] and [9]. As the CTL semantics are defined over Kripke structures, we will also call these structures CTL models.

2.1 Kripke Modal Transition System

The Kripke Modal Transistion System is capable of representing incomplete system information explicitly. A KMTS has two kinds of transitions, must transitions and may transitions, that express transitions that must occur (certain behaviour) and transitions that may occur (the behaviour is uncertain) in the system. Incomplete information can be also expressed in the states of a KMTS, because in any state of the model an atomic property can be defined or undefined (uncertain state).

Definition 3. *Let* AP *be a set of atomic propositions and* $Lit = AP \cup \{\neg p \mid p \in AP\}$ *the set of literal over* AP*. A Kripke modal transition system (KMTS) is a tuple* $M = (AP, S, R^+, R^-, L)$*, where* S *is a set of finite sates,* $R^+ \subseteq S \times S$ *and* $R^- \subseteq S \times S$ *are transition relations such that* $R^+ \subseteq R^-$*, and* $L : S \to 2^{Lit}$ *is a label function, such that for all state* s *and* $p \in AP$*, at most one between* p *and* $\neg p$ *occurs.*

In the definition above the transitions R^+ and R^- correspond to the transitions *must* and *may* respectively.

[1] Although the CTL semantics consider Kripke structures with total relation transition, such a requirement can be released and we assume a Kripke structure with a partial transition relation instead.

2.2 KMTS as a set of Kripke Structures

In [6] the authors interpret a KMTS as a set of Kripke structures. According to this interpretation, a (s_i, s_j) *may* transition can lead to two CTL models, one with a (s_i, s_j) transition and another without this transition, and an indefinite literal l in a state s_i can lead to two CTL models: one which has l labelled in the correspondent state s_i and another one with $\neg l$ labelled in s_i. The authors define a KMTS expansion in CTL models with respect to all their indetermination leading to an exponential set with 2^m Kripke structures, where m is the number of indeterminations.

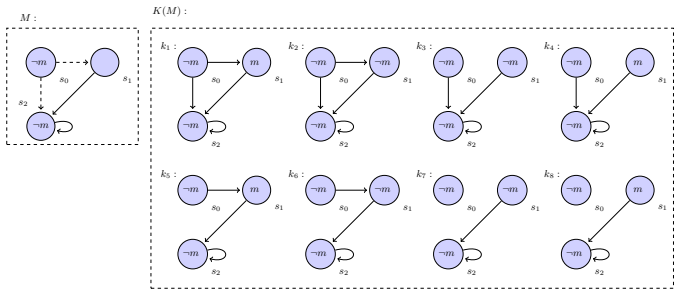


Fig. 1: Expansion K(M) of the KMTS M. The dashed arrows represent *may* transitions and the solid ones represent *must* transitions.

Let M be a KMTS, we denote by $K(M)$ the set of Kripke structures represented by M. Figure 1 illustrates a KMTS M and its expansion set $K(M)$. Since the state s_1 of M is neither labelled with m nor $\neg m$, and the KMTS has two *may* transitions ($s_0 \rightarrow s_1$ and $s_0 \rightarrow s_2$), M leads to eight CTL models.

Definition 4. *Let M_1 and M_2 be two KMTSs. We say M_1 is an instance of M_2 denoting by $M_1 \sqsubseteq M_2$ iff $K(M_1) \subseteq K(M_2)$.*

An instance of a KMTS M represents a subset of Kripke structures from K(M) and in order to represent this subset, an instance of M can be addressed. To do so, we use some change operations that change a KMTS M into an instance of it. We argue that sets of instances represented by change operations have some desired properties which allow the definition of some set operations, defined in Section 4, such as the contraction operation which are the key for the contraction model checking.

There are 3 primitive change operations to generate KMTS instances:

P_1 (s, s'): which removes the pair (s, s') from the relation R^-
P_2 (s, s'): which transforms (s, s') of R^- to (s, s') of R^+
P_3 (s, l): which assigns a literal l to the state s if l is undefined in $L(s)$

As for example, the application of the operation $P_2(s_0, s_2)$ over the KMTS M presented in Figure 1 generates a KMTS that exactly represents the Kripke structures k_1, k_2, k_3 and k_4 presented in the same figure. Likewise, the application of the operations $P_2(s_0, s_2)$ and $P_3(s_1, \neg m)$ generates a KMTS that exactly represents the Kripke structures k_2 and k_3.

The application of a change over a KMTS is sometimes not encouraged. For example, we cannot apply the operations $P_3(s, l)$ and $P_3(s, \neg l)$ over a KMTS nor can we apply $P_1(s, s')$ and $P_2(s, s')$ because we cannot define a KMTS that has a transition $s \to s'$ that belongs to the set R^+, but it does not belong to the set R^-. We call such operations complement operations.

Definition 5. *Let p a primitive operation, the complement operation of p denoted by $\neg p$ is defined as follows:*

(i) $p = P_3(s, l)$ *iff* $\neg p = P_3(s, \neg l)$
(ii) $p = P_1(s, s')$ *iff* $\neg p = P_2(s, s')$

We say that two set of changes are not compatible if both have at least a complement primitive operation of the another one.

Definition 6. *A set X of changes is not compatible with a set Y of changes, denoting by $X \not\simeq Y$, iff exists an operation $p \in X$ such that $\neg p \in Y$.*

3 Semantics of CTL with respect to KMTS

In this section we consider some notations, definitions and properties used to define the semantics of CTL for KMTS interpreted as a set of Kripke structures which is presented at the end of this section.

Definition 7. *Let M be a KMTS. The set of M states reachable from a state $s \in S$ of M is the set $\overrightarrow{S}(s) = \{s' \in S | s \to s' \in R^-\}$.*

Definition 8. *Let M be a KMTS, $s \to s' \in R^-$ a may transition of it. The subset of Kripke structures in $K(M)$ that does not have the transition $s \to s'$ is the set $\left[R/\right]_M(s, s') = \{k \in K(M) \mid s \to s' \notin k\}$.*

Definition 9. *Let M be a KMTS and φ a CTL formula. The subset of Kripke structures represented by M that satisfies φ starting at a state $s \in S$ is the set $\left[\varphi\right]_M^s = \{k \in K(M) \,|k, s \models \varphi\}$.*

Definition 10. *Let M be a KMTS and φ a CTL formula. We define the sets successive union($\left[\bigcup\right]_M^s(\varphi)$) and successive intersection ($\left[\bigcap\right]_M^s(\varphi)$) as follows:*

$$\left[\bigcup\right]_M^s(\varphi) = \bigcup_{s' \in \overrightarrow{S}(s)} \left(\left[\varphi\right]_M^{s'} \setminus \left[R/\right]_M(s, s')\right);$$

$$\left[\bigcap\right]_M^s(\varphi) = \bigcap_{s' \in \overrightarrow{S}(s)} \left(\left[\varphi\right]_M^{s'} \cup \left[R/\right]_M(s, s')\right).$$

The set successive union captures all the Kripke structures represented by M that satisfy φ starting at a state s' reachable from a state s through the transition $s \to s'$, excluding those ones that do not have such a transition. On the other hand, the set successive intersection captures all Kripke structures that satisfy φ starting at every state s' reachable from s through the transition $s \to s'$. Indeed, the sets successive union and successive intersection capture those models that satisfy the CTL formulas $EX\varphi$ and $AX\varphi$, respectively.

Proposition 1. *Let M be a KMTS, s a state of it and φ a CTL formula. A Kripke structure k belongs to $\left[\bigcup\right]^s_M(\varphi)$ iff $k \in K(M)$ and $k, s \models EX\varphi$.*

Proof. $k, s \models EX\varphi$ *iff* $\exists s \to s' \in k$ *s.t* $k, s' \models \varphi$ *(I).*

"⇒" *From Definition 9, we have* $\forall k \in \left[\varphi\right]^s_M$; $k, s \models \varphi$ *and* $k \in K(M)$. *So,* $\forall k \in \left(\left[\varphi\right]^{s'}_M \setminus \left[R/\right]_M(s, s')\right)$; $k, s' \models \varphi$ *and* $s \to s' \in k$ *(II). Thus, from (II) and (I), we have* $\forall k \in \left(\left[\varphi\right]^{s'}_M \setminus \left[R/\right]_M(s, s')\right)$; $k, s \models EX\varphi$. *This way,* $\forall k \in \bigcup\limits_{s' \in \overrightarrow{S}(s)} \left(\left[\varphi\right]^{s'}_M \setminus \left[R/\right]_M(s, s')\right)$; $k, s \models EX\varphi$ *and* $k \in K(M)$. *Therefore,* $\forall k \in \left[\bigcup\right]^s_M(\varphi)$; $k, s \models EX\varphi$ *and* $k \in K(M)$.

"⇐" *Let us suppose* $k, s \models EX\varphi$ *and* $k \in K(M)$. *From* (I), $\exists s \to s' \in k$, $k, s' \models \varphi$. *Thus,* $k \in \left[\varphi\right]^{s'}_M$ *and* $k \notin \left[R/\right]_M(s, s')$. *So,* $k \in \left[\varphi\right]^{s'}_M \setminus \left[R/\right]_M(s, s')$. *Therefore,* $k \in \bigcup\limits_{s' \in \overrightarrow{S}(s)} \left[\varphi\right]^{s'}_M \setminus \left[R/\right]_M(s, s')$ *and* $k \in \left[\bigcup\right]^s_M(\varphi)$.

Definition 11. *The semantics of a CTL formula φ in its negation normal form w.r.t a KMTS is presented in Table 1.*

Table 1: Semantics of a CTL formula φ w.r.t a KMTS M.

Formula	$\top$	F	$\bot$
$\|\|l\|\|_M(s)$	$l \in L(s)$	$\neg l \in L(s)$	otherwise
$\|\|\varphi_1 \wedge \varphi_2\|\|_M(s)$	$\left[\varphi_1\right]^s_M \cap \left[\varphi_2\right]^s_M = K(M)$	$\left[\varphi_1\right]^s_M \cap \left[\varphi_2\right]^s_M = \emptyset$	otherwise
$\|\|\varphi_1 \vee \varphi_2\|\|_M(s)$	$\left[\varphi_1\right]^s_M \cup \left[\varphi_2\right]^s_M = K(M)$	$\left[\varphi_1\right]^s_M \cup \left[\varphi_2\right]^s_M = \emptyset$	otherwise
$\|\|EX\varphi\|\|_M(s)$	$\left[\bigcup\right]^s_M(\varphi) = K(M)$	$\left[\bigcup\right]^s_M(\varphi) = \emptyset$	otherwise
$\|\|AX\varphi\|\|_M(s)$	$\left[\bigcap\right]^s_M(\varphi) = K(M)$	$\left[\bigcap\right]^s_M(\varphi) = \emptyset$	otherwise
$\|\|E[\varphi_1 U \varphi_2]\|\|_M(s)$	$\left[\varphi_2\right]^s_M \cup \left(\left[\varphi_1\right]^s_M \cap \left[\bigcup\right]^s_M(E[\varphi_1 U \varphi_2])\right) = K(M)$	$\left[\varphi_2\right]^s_M \cup \left(\left[\varphi_1\right]^s_M \cap \left[\bigcup\right]^s_M(E[\varphi_1 U \varphi_2])\right) = \emptyset$	otherwise
$\|\|A[\varphi_1 U \varphi_2]\|\|_M(s)$	$\left[\varphi_2\right]^s_M \cup \left(\left[\varphi_1\right]^s_M \cap \left[\bigcup\right]^s_M(A[\varphi_1 U \varphi_2])\right) = K(M)$	$\left[\varphi_2\right]^s_M \cup \left(\left[\varphi_1\right]^s_M \cap \left[\bigcup\right]^s_M(A[\varphi_1 U \varphi_2])\right) = \emptyset$	otherwise
$\|\|E[\varphi_1 R \varphi_2]\|\|_M(s)$	$\left[\varphi_2\right]^s_M \cap \left(\left[\varphi_1\right]^s_M \cup \left[\bigcup\right]^s_M(E[\varphi_1 U \varphi_2])\right) = K(M)$	$\left[\varphi_2\right]^s_M \cap \left(\left[\varphi_1\right]^s_M \cup \left[\bigcup\right]^s_M(E[\varphi_1 U \varphi_2])\right) = \emptyset$	otherwise
$\|\|A[\varphi_1 R \varphi_2]\|\|_M(s)$	$\left[\varphi_2\right]^s_M \cap \left(\left[\varphi_1\right]^s_M \cup \left[\bigcup\right]^s_M(A[\varphi_1 U \varphi_2])\right) = K(M)$	$\left[\varphi_2\right]^s_M \cap \left(\left[\varphi_1\right]^s_M \cup \left[\bigcup\right]^s_M(A[\varphi_1 U \varphi_2])\right) = \emptyset$	otherwise

Proposition 2. *Let M be a KMTS and φ a CTL formula, then*

$$\|\varphi\|_M(s) = \begin{cases} \top \ \textit{iff } \forall k \in K(M); k, s \models \varphi = T; \\ F \ \textit{iff } \forall k \in K(M); k, s \models \varphi = F; \\ \bot, \textit{otherwise} \end{cases}$$

Proof. It follows straight from the semantics.

4 KMTS Operations

The CTL semantics over KMTS interpreted as a set of Kripke structure deals directly with set operations. Thus, in order to decide if a set of Kripke structures represented by a KMTS M satisfies a CTL property, we should decide if every CTL model in $K(M)$ satisfies such a property. However, it is not necessary because we can deal directly with M, instead of K(M). Therefore, in this section, we define set operations over KMTSs and we prove some properties and some limitations as well.

We write $M(X)$ to denote an instance generated by the application of a set X of changes over a KMTS M.

Definition 12. *Let M, M_1, M_2 be KMTSs, X_1 and X_2 two set of changes, such that $M_1 \sqsubseteq M, M_2 \sqsubseteq M$ and $M_1 = M(X_1), M_2 = M(X_2)$. We define the operations intersection, union and difference, with respect to KMTS as :*

Union: $M_1 \sqcup M_2 = \{M_1, M_2\}$

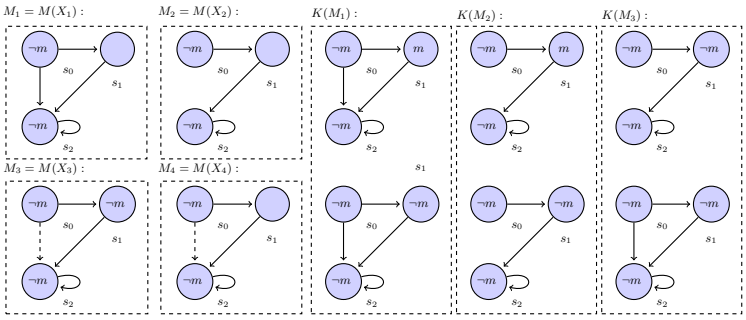
Intersection: $M_1 \sqcap M_2 = \begin{cases} \emptyset & , \textit{ iff } X_1 \not\simeq X_2 \\ \{M(X_1 \cup X_2)\} & , \textit{otherwise} \end{cases}$

Difference: $M_1 \setminus M_2 = \begin{cases} \{M_1\} & \textit{iff } X_1 \not\simeq X_2 \\ \bigcup\limits_{p_i \in (X_2 \setminus X_1)} \{M\big(X_1 \cup \{\neg p_i\}\big)\} & \textit{otherwise} \end{cases}$

The difference operation $M_1 \setminus M_2$ generates a set of KMTS such that the CTL models represented by them are present in $K(M_1)$ but are not present in $K(M_2)$. As M_1 and M_2 are instances of M, i.e., they can be defined from changes over M then the models resulting from the difference can be defined from the set of changes that generate M_1 and M_2, i.e., X_1 and X_2, respectively. As a result, if X_1 and X_2 are not compatible the intersection between the models is empty and the difference is M_1. If they are compatible, then the models resulting from the difference are obtained from M by the set of changes X_1 (X_1 generates M_1 from M) together with the complementary primitive operations in X_2 which do not belong to X_1 ($X_1 \cup \{\neg p_i\}$) to eliminate the intersection between $K(M_1)$ and $K(M_2)$. For example, for the KMTS M_2 and M_3 illustrated in Figure 2, $M_2 \setminus M_3$ generates a set containing only the top Kripke structure of the set $K(M_2)$. This model is defined by the set of changes $X' = \{P_1(s_0, s_2),\ P_2(s_0, s_1)\} \cup \{P_3(s_1, m)\}$ since

$M_2 = M(\{P_1(s_0, s_2),\ P_2(s_0, s_1)\})$ and $M_3 = M(\{P_2(s_0, s_1), P_3(s_1, \neg m)\})$ and $P_3(s_1, \neg m)$ does not belong to X_1.

The intersection operation $M_1 \sqcap M_2$ generates a single set with a KMTS that only represents the CTL models in $K(M_1) \cap K(M_2)$. The union operation is simple and no further explanation is needed.



$X_1 = \{P_2(s_0, s_1), P_2(s_0, s_2)\}$ $\quad$ $X_2 = \{P_2(s_0, s_1), P_1(s_0, s_2)\}$
$X_3 = \{P_2(s_0, s_1), P_3(s_1, \neg m)\}$ $\quad$ $X_4 = \{P_2(s_0, s_1)\}$

Fig. 2: Instances M_1, M_2, M_3 and M_4 from the KMTS M illustrated in the Fig. 1 and the expansion sets $K(M_1), K(M_2)$ and $K(M_3)$.

Sometimes the set of Kripke structures represented by two KMTS can be represented by a single KMTS. For example, the expansion set of the KMTS M_4 in Figure 2 is exactly the union of the expansion set of the KMTSs M_1 and M_2, i.e., $K(M_4) = K(M_1) \cup K(M_2)$, as a result $\{M_1, M_2\}$ can be expressed by a single KMTS, which is in this case M_4. In this sense, we define a contraction operation which is a specific kind of union for these cases. If two KMTSs cannot be contracted into a single one, then the contraction operation will be equivalent to the union operation.

Definition 13. *Let M be a KMTS, M_1 and M_2 instances of M generated, respectively, by the set of changes X_1 and X_2. The contraction operation, denoted by $M_1 \sqcup^+ M_2$, is defined as:*

$$M_1 \sqcup^+ M_2 = \begin{cases} \{M(X_1 \cap X_2)\} & \textit{iff } X_1 \subseteq X_2 \textit{ or } X_2 \subseteq X_1 \textit{ or} \\ & \exists p \in X_1 \textit{ s.t } \neg p \in X_2 \textit{ and} \\ & X_1 \setminus \{p\} = X_2 \setminus \{\neg p\} \\ \{M_1,\ M_2\} & \textit{otherwise} \end{cases}$$

In order to contract two models into a single one, it is necessary (but not enough) that both models have at most one complementary operation w.r.t each

other, otherwise the contraction operation results in a set containing exactly the two input models. We explain the contraction operation through an example. Let us suppose KMTSs $M_1 = M(X_1)$ and $M_2 = M(X_2)$ such that $X_1 \subseteq X_2$. Thus, $X_1 \cap X_2 = X_1$ and $K(M_2) \subseteq K(M_1)$ which implies that the contraction results in M_1, i.e., $M_1 \sqcup^+ M_2 = M(X_1 \cap X_2) = M_1$. Consider now X_1 and X_2 such that X_1 has among all its operations only one operation p which is complementary with an operation $\neg p$ in X_2 and X_1 is equal to X_2 unless this operation. The operation p or $\neg p$ can be applied over M leading to the instances $M_1 = M(X_1)$ and $M_2 = M(X_2)$. So, this case reduces to the previous one, i.e., the contraction result is $M(X_1 \cap X_2)$ since $K(M(X_1 \cap X_2)) = K(M_1) \cup K(M_2))$.

In Figure 2, the model M_1 can be contracted with M_2, i.e., $M_1 \sqcup^+ M_2 = M_4$ because $X_1 \setminus \{P_2(s_0, s_2)\} = X_2 \setminus \{P_1(s_0, s_2)\}$ and $P_1(s_0, s_2)$ is complementary with $P_2(s_0, s_2)$. Furthermore, the model M_1 can be contracted with M_4 and the model M_2 can be contracted with M_4, both resulting in the model M_4. However, M_3 cannot be contracted with any other model in Figure 2.

4.1 Dealing with Sets of KMTSs

To deal with a set of CTL models, we can consider a KMTS whose expansion represents these models. However, a single KMTS sometimes is not capable of representing a specific set of CTL models and in order to perform such a task a set of KMTSs can be addressed. Computationally, a set of KMTSs is far more convenient because it is preferable to deal directly with a KMTS instead of its expansion set. In order to achieve our contraction model checking we define in this section a partition set and a full partition set over a set of KMTSs and we define some operations over them.

Let M be a KMTS and Γ a set of instances of M. It is always possible to construct a set Γ_p such that each element in it is an instance of M and the intersection of any two instances of Γ is always empty.

Definition 14. *Let M be a KMTS and Γ a set of instances of it. Γ is a Partition Set (PS) of M iff every model in Γ is an instance of M and $\forall M_1, M_2 \in \Gamma, M_1 \sqcap M_2 = \emptyset$.*

Definition 15. *Let Γ and Γ' be two set of instances of a KMTS M. Γ and Γ' are equivalent, denoting by $\Gamma \equiv \Gamma'$, iff $\bigcup_{M_i \in \Gamma} K(M_i) = \bigcup_{M_i \in \Gamma'} K(M_i)$.*

Definition 16. *Let Γ_1 and Γ_2 be two sets of instances of a KMTS. We define the difference ($\setminus\!\!\setminus$), intersection ($\sqcap$) and union ($\sqcup$) operations as:*

$$\Gamma_1 \setminus\!\!\setminus \Gamma_2 = \bigcup_{\substack{M_i \in \Gamma_1, \\ M_k \in \Gamma_2}} M_i \setminus M_k \qquad \Gamma_1 \sqcap \Gamma_2 = \bigcup_{\substack{M_i \in \Gamma_1, \\ M_k \in \Gamma_2}} M_i \sqcap M_k$$

$$\Gamma_1 \sqcup \Gamma_2 = \Gamma_1 \cup PS(\Gamma_2 \setminus\!\!\setminus (\Gamma_1 \sqcap \Gamma_2))$$

where $PS(\Gamma)$ is a Partition Set equivalent to a set of instances Γ.

Let $K(\Gamma)$ be the set of all Kripke structures represented by every KMTS in Γ. The difference, intersection and union operation in Definition 16 calculates respectively the difference, intersection and union of the set with models sets represented by these KMTSs. In relation to the intersection operation defined over two KMTSs sets Γ_1 and Γ_2 it results in a set Γ' such that $K(\Gamma') = K(\Gamma_1) \cap K(\Gamma_2)$. The union and difference operations are interpreted similarly.

Proposition 3. *If Γ_1 and Γ_2 are two PS of a KMTS M, then $\Gamma_1 \sqcap \Gamma_2$ and $\Gamma_1 \sqcup \Gamma_2$ is a PS.*

Proof. It follows straight from the Definition 16.

Theorem 1. *For any set of instances Γ of a KMTS M there is always a PS Γ' such that $\Gamma \equiv \Gamma'$.*

Proof. Let M be a KMTS, Γ a set of instances of M and M_1 a M instance in Γ. Create the set $\Gamma_1 = \Gamma \setminus\setminus \{M_1\}$. Then, $\forall M_i \in \Gamma_1, M_i \sqcap M_1 = \emptyset$ by the difference operation and $\Gamma_1 \cup \{M_1\} \equiv \Gamma$. Choose an element $M_2 \in \Gamma_1$, then the set $\Gamma_1' = \{M_1, M_2\}$ is a PS. Create now the set $\Gamma_2 = \Gamma_1 \setminus\setminus \{M_2\}$, then $\forall M_i \in \Gamma_2,\ M_i \sqcap M_2 = \emptyset$ and $M_i \sqcap M_1 = \emptyset$ and $\Gamma_2 \cup \{M_1, M_2\} \equiv \Gamma$. Choose an element M_3 in Γ_2, then $\{M_1, M_2, M_3\}$ is a PS and $\Gamma_3 \cup \{M_1, M_2, M_3\} \equiv \Gamma$. Following this construction we achieve at the end a PS equivalent to Γ.

If a PS of a KMTS M represents all the CTL models of K(M), then we say such a PS is a Full Partition Set.

Definition 17. *Let M be a KMTS and Γ a set of instances of it. We say Γ is a Full Partition Set (FPS) of M iff Γ is a PS and $K(M) = \bigcup_{M_i \in \Gamma} K(M_i)$.*

Corollary 1. *If Γ is a set of instances of a KMTS M and $K(M) = \bigcup_{M_i \in \Gamma} K(M_i)$, then there is a FPS Γ' such that $\Gamma \equiv \Gamma'$.*

Every KMTS M can be obtained from a finite number of contraction operations over a FPS of M. In order to prove it, we first define a Tree Partition Set and show how to represent a PS over this structure.

4.2 Tree Partition Set

A Tree Partition Set (TPS) is a binary tree that represents a partition set Γ from a KMTS M. Each node v of a TPS is labelled by a primitive change operation $L_T(v)$ applicable over M.

Definition 18. *A Tree Partition Set (TPS) of a KMTS M is a tuple $T_M = (N, v_0, E, L_T, Lf, Rg)$, where N is a finite set of nodes, $v_0 \in N$ is the root node, $E \subseteq N \times N$ is the set of edges, L_T is a partial labelling function that maps each node in N to a primitive operation applicable over M such that v_0 is the only node of a TPS that is not defined in L_T; and Lf, Rg are partial functions that map a node to its left and right child respectively. Furthermore, for every non-end node $v \in N$, there are nodes $v_1, v_2 \in N$ such that $Lf(v) = v_1$ and $Rg(v) = v_2$ iff $L_T(v_1) = p$ and $L_T(v_2) = \neg p$.*

Let $\pi = v_0 \to v_1 \to \cdots \to v_n$ be a path between the nodes v_0 and v_n in a TPS, we denote by $\pi \setminus v_0$ the subpath $v_1 \to \cdots \to v_n$ of π, and $v \in \pi$ to denote that a node v belongs to a path π. We highlight that each operation that labelled a vertice along a path v_0 to v_n follows from an indetermination of the KMTS.

Definition 19. *Let M a KMTS, $T_M = (N, v_0, E, L_T, Lf, Rg)$ a TPS. We define the operation $Change(v_n)$ that maps the single path $\pi = v_0 \to v_1 \to \cdots \to v_n$ between v_0 and any node $v_n \in T_M$ to a set of changes applicable over M as:*

$$Change(v_n) = \bigcup_{v_k \in \pi \setminus v_0} \{L_T(v_k)\}$$

A change from a node v as defined above considers all primitive changes that occur along the single path from the root to v excluding the root node.

The $Change(v)$ of any node in a TPS T_M when applied over a KMTS M generates an instance of it and we say a node represents an instance of M. The set of instances represented by every end-node of a TPS is a PS.

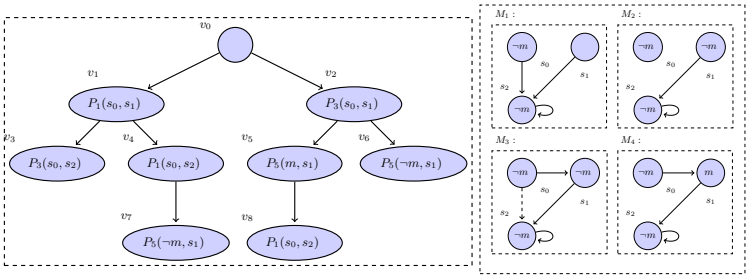


Fig. 3: A Tree Partition Set that represents the PS $\{M_1, M_2, M_3, M_4\}$.

The Figure 3 illustrates a PS Γ from the KMTS M presented in Figure 1 and a TPS T_M that represents Γ. M_1, M_2, M_3 and M_4 are generated from changes of TPS end-nodes, i.e., by the application of $Change(v_3), Change(v_7), Change(v_6)$ and $Change(v_8)$ over M, respectively .

Lemma 1. *Let Γ be a PS w.r.t a KMTS M represented by TPS T_M and M' an instance of M. If $\Gamma \cup \{M'\}$ results in a PS, then there is a PS Γ' equivalent to it that can be also represented by a TPS.*

Proof. From the contraction operation we have that for any set of change X and primitive operation p, $M(X \cup \{p\} \sqcup^+ M(X \cup \{\neg p\}) = M(X)$ (I). Create a TPS T'_M equal to T_M. If M' is an instance of M then it is generated from a set X' of changes, i.e., $M' = M(X')$. Select in T'_M the following subpath $\pi = v_0 \to \cdots \to v_i$ in such a way $Change(v_i) \subset X'$ and v_i has only one child v_{i+1} such that $L_T(v_{i+1}) \notin X'$ or v_i has two children where $L_T(Lf(v_i)) \notin X'$ and $L_T(Rg(v_i)) \notin X'$.

Case 1. *v_i has only one child v_{i+1} and $L_T(v_{i+1}) \notin X'$. Then there are two cases: $\neg L_T(v_{i+1}) \in X'$ or $\neg L_T(v_{i+1}) \notin X'$.*
(a) $\neg L_T(v_{i+1}) \in X'$. Create a node v_{i+2} to be the other child of v_i and add to T'_M the following path $\pi_1 = v_0 \to \cdots \to v_i \to v_{i+2} \to \cdots \to v'$ such that $Change(v')$ $= X'$ and $L_T(v_{i+2}) = \neg L_T(v_{i+1})$. Thus, the resulting T'_M represents the PS $\Gamma' = \Gamma \cup \{M'\}$.
(b) $\neg L_T(v_{i+1}) \notin X'$. Let $X_i = Change(v_i)$, then $X_i \subset X'$ and $X' = X_i \cup (X' \setminus X_i)$. Let $M'_1 = M(X' \cup \{p\})$ and $M'_2 = M(X' \cup \{\neg p\})$, where $p = L_T(v_{i+1})$. From (I) we have $M' = M(X') = M(X' \cup \{p\}) \sqcup^+ M(X' \cup \{\neg p\})$ which means $M' = M'_1 \sqcup^+ M'_2$. Thus, the set $\Gamma' = \Gamma \cup \{M'_1, M'_2\}$ and it is equivalent to $\Gamma \cup \{M'\}$ and by hypothesis it is a PS. So, if we represent M'_1 and M'_2 in T'_M then we achieve a representation of M' in T'_M. To do so, create a node v_{i+2} to be the other child of v_i and add to T'_M the path $\pi_1 = v_0 \to \ldots v_i \to v_{i+2} \to \cdots \to v'$ such that $L_T(v_{i+2}) = \neg L_T(v_{i+1})$ and $Change(v') = X' \cup \{L_T(v_{i+2})\}$. Thus T'_M represents M'_2. Let $X'_1 = X' \cup \{L_T(v_{i+1})\}$, in order to represent M'_1, we must create a path $\pi_2 = v_0 \to \cdots \to v_i \to v_{i+1} \to \cdots \to v'$ such that $Change(v') = X'_1$. To achieve this select in T'_M a subpath $\pi'_2 = v_0 \to \cdots \to v_i \to v_{i+1} \to \cdots \to v_k$ in such a way $Change(v_k) \subset X'_1$ and or v_k has only one child v_{k+1} such that $L_T(v_{k+1}) \notin X'_1$ or v_k has two children where $L_T(Lf(v_k)) \notin X'_1$ and $L_T(Rg(v_k)) \notin X'_1$. This reduces to the cases (1) and (2).

Case 2. *v_i has two children where $L_T(Lf(v_i)) \notin X'$ and $L_T(Rg(v_i)) \notin X'$. Let $X_i = Change(v_i)$, then $X_i \subset X'$ and $X' = X_i \cup (X' \setminus X_i)$. Let $M'_1 = M(X' \cup \{p\})$ and $M'_2 = M(X' \cup \{\neg p\})$, where $p = L_T(Lf(v_1))$. From (I) we have $M' = M(X') = M(X' \cup \{p\}) \sqcup^+ M(X' \cup \{\neg p\})$ which means $M' = M'_1 \sqcup^+ M'_2$. Thus, the set $\Gamma' = \Gamma \cup \{M'_1, M'_2\}$ is is equivalent to $\Gamma \cup \{M'\}$ and by hypothesis it is a PS. So, we must represent in T'_M the instances M'_1 and M'_2. Let $X'_1 = X' \cup \{L_T(Lf(v_i))\}$ and $X'_2 = X' \cup \{L_T(Rg(v_i))\}$, in order to represent M'_1 we must create a path $\pi_1 = v_0 \to \cdots \to v_i \to Lf(v_i) \to \cdots \to v_j$ where $Change(v_j) = X'_1$. Select in T'_M a subpath $\pi'_1 = v_0 \to \cdots \to v_i \to Lf(v_i) \to \cdots \to v_k$ in such a way $Change(v_k) \subset X'_1$ and or v_k has only one child v_{k+1} wherein $L_T(v_{k+1}) \notin X'_1$, or v_k has two children where $L_T(Lf(v_k)) \notin X'_1$ and $L_T(Rg(v_k)) \notin X'_1$. This reduces to the cases (1) and (2). To represent M_2 we proceed simillary as we have done to M_1. Since a path in a TPS is finite, eventually the cases $1-(b)$ and 2 will lead to the case $1-(a)$ and then the TPS T'_M represents the resulting PS Γ'.*

From Lemma 1, we prove that for every PS Γ there is always a PS Γ' equivalent to Γ that is represented by a TPS.

Theorem 2. *If Γ is a PS w.r.t a KMTS M, then there is always a PS Γ' equivalent to Γ which can represented by a TPS T_M.*

Proof. Let $\Gamma = \{M_1, \ldots, M_n\}$, each $\{M_i\}$ is also a PS by definition. From Lemma 1, there is a PS $\Gamma_2 \equiv \{M_1\} \cup \{M_2\}$ which can be represented by a TPS. In addition, a PS Γ_3 equivalent to $\Gamma_2 \cup \{M_3\}$ can also be generated and can be represented by a TPS as well. Sucessively, a PS $\Gamma_n \equiv \Gamma_{n-1} \cup \{M_n\}$ that can be represented by a TPS can be generated which is equivalent to $\{M_1, \ldots, M_n\} = \Gamma$.

Theorem 3. *Let M be a KMTS and $T_M = (N, v_0, E, L_T, Lf, Rg)$ a TPS that represents a PS Γ defined from M. If Γ is a FPS then for every non-end node $v_k \in N$ and $X_k = Change(v_k)$ $M_k = M(X_k)$ can be generated from a finite number of contraction operations in Γ.*

Proof. The proof is by induction on the structure of T_M. Let $\pi_k = v_0 \to \cdots \to v_k$ be a path in T_m. If M has m indefinites, then $1 \leq k \leq m$.

Base case: *let $\pi_k = v_0 \to \cdots \to v_{k-1} \to v_k$ be a longest path of T_M.Then, v_k is an end-node. As π_k is one of the longest path and Γ is a FPS, then the path $\pi'_k = v_0 \to \cdots \to v_{k-1} \to v'_k$ belongs to T_M, where $L_T(v_k) = \neg L_T(v'_k)$. Let $p = L_T(v_k)$, $X_{k-1} = Change(v_{k-1})$. $Change(v_k) = X_{k-1} \cup \{p\}$ and $Change(v'_k) = X_{k-1} \cup \{\neg p\}$. So we have $M(X_{k-1}) = M(X_{k-1} \cup \{p\}) \sqcup^+ M(X_{k-1} \cup \{\neg p\})$. Therefore, $M(X_{k-1})$ is obtained by one contraction operation in Γ.*

Induction Step: *let us suppose any $M(X_k)$ can be generated by a finite number of contraction operations. We show that $\forall k-1$, $M(X_{k-1})$ can be generated by a finite number of contraction operations as well. The path $\pi_{k-1} = v_0 \to \cdots \to v_{k-1}$ has two children v_k and v'_k where $L_T(v_K) = \neg L_T(v'_k)$, due to the fact Γ is a FPS. Let $p = L_T(v_k)$, $X_{k-1} = Change(v_{k-1})$, $X_k = Change(v_k)$ and $X'_k = Change(v'_k)$. Then, $X_k = X_{k-1} \cup \{p\}$ and $X'_k = X_{k-1} \cup \{\neg p\}$. So, $M(X_{k-1}) = M(X_k) \sqcup^+ M(X'_k)$. By the induction hypothesis $M(X_k)$ and $M(X'_k)$ can be obtained by a finite number of contraction operations and so is $M(X_{k-1})$.*

Theorem 3 implies in the Corollary 2 which guarantees that if we have a FPS of M then it is always possible to produce M from a set of contractions operations over this FPS. The contraction model checking algorithm uses contractions operations over partitions sets and when a full partition set is achieved it means that all Kripke models satisfy the property been verified and consequently M represents the truth values $\top$. The other truth values are represented by PS that is not a FPS and in this case if the PS is not empty then it represents the truth values $\bot$ and if is empty it represents the truth values F.

Corollary 2. *Let M be a KMTS and Γ a PS of it. If Γ is a FPS then M can be generated from a finite number of contraction operations in Γ.*

Let Γ be a PS and Γ' a PS resulting from a finite number of contraction operations over Γ, we say Γ' is a Maximal Partition Set (MPS) if there is no more contraction applicable over Γ. Therefore, according to Corollary 2 if a PS is in fact a FPS then its MPS is exactly the single set that contains M. We write $\sqcup^+(\Gamma)$ to denote the MPS resulting from Γ.

5 The Contraction Model Checking

Some works such as [11] and [5] propose model checking game approaches, which consist of a game played over a board by two players: $\exists$ve and $\forall$belard. The

former tries to prove that the CTL specification holds, whereas the latter tries to refute the specification. In order to decide the winner of the game (respectively the model checking result), a colouring algorithm that maps each configuration of the game board to one truth value is defined. The colour set for the initial configuration of the board is the model checking result. A model checking game combined with the contraction operation applied in each configuration of the game suffices to determine the model checking result to a KMTS interpreted as a set of CTL models. We call this new approach Contraction Model Checking.

Let M be a KMTS, s_0 a state of it and φ a CTL formula. The model checking game for $M, s_0 \models \varphi$ is played over a board (game-graph) constructed according to the game rules that define the possible moves each player can make in a configuration it owns. A board is a graph of configurations constructed by decomposing φ in its subformulas following the game rules presented in Figure 4. Every configuration of a game-graph belongs to $S \times sub(\varphi)$, where S is the set of M states and $sub(\varphi)$ is the set of subformulas of φ. We denote a configuration by $s \vdash \psi$, where ψ is a subformula of φ and s is a state of M.

(1) $\frac{s \vdash \psi_0 \vee \psi_1}{s \vdash \psi_i} : i \in \{0,1\}$ $(\exists ve)$	(2) $\frac{s \vdash \psi_0 \wedge \psi_1}{s \vdash \psi_i} : i \in \{0,1\}$ $(\forall belard)$
(3) $\frac{s \vdash EX\,\varphi}{t \vdash \varphi} : (s,t) \in R^-$ $(\exists ve)$	(4) $\frac{s \vdash AX\,\varphi}{t \vdash \varphi} : (s,t) \in R^-$ $(\forall belard)$
(5) $\frac{A(\varphi_1 U \varphi_2)}{s \vdash \varphi_2 \vee (\varphi_1 \wedge AXA(\varphi_1 U \varphi_2))} (\exists ve)$	(6) $\frac{E(\varphi_1 U \varphi_2)}{s \vdash \varphi_2 \vee (\varphi_1 \wedge EXE(\varphi_1 U \varphi_2))} (\exists ve)$
(7) $\frac{A(\varphi_1 R \varphi_2)}{s \vdash \varphi_2 \wedge (\varphi_1 \vee AXA(\varphi_1 R \varphi_2))} (\exists ve)$	(8) $\frac{E(\varphi_1 R \varphi_2)}{s \vdash \varphi_2 \wedge (\varphi_1 \vee EXE(\varphi_1 R \varphi_2))} (\exists ve)$

Fig. 4: Game rules for the Model Checking Game

Let G be the game-graph for a KMTS M and CTL formula φ. The contraction model checking is a colouring function $\chi : V \to \{\top, F, \bot\}$, where V is the set of vertices of G, that maps each configuration in G to one truth value. We can observe that the final truth value is calculated from sets of KMTS models that represents truth values in the model checking process. The colouring function is defined over a maximal contraction function δ that maps each configuration of the game-graph to a maximal PS Γ. The expansion of the KMTSs in this PS is exactly the set of Kripke structures that satisfy the formula in the respectively state. As the resulting PS Γ is a MPS, $\Gamma = \{M\}$ iff M satisfies φ in the respectively input configuration, $\Gamma = \emptyset$ if no CTL model in $K(M)$ satisfies φ, and Γ is a PS different from $\{M\}$ and $\emptyset$ otherwise.

Definition 20. *Let M be a KMTS, s and s' states of M, φ a CTL formula and G the board of the game for the model checking $M, s_0 \models \varphi$. The maximal contraction function δ is defined recursively as:*

$$\delta(s \vdash l) = \begin{cases} \{M\} & \textit{iff } l \in L(s); \\ \emptyset & \textit{iff } \neg l \in L(s); \\ \{M(\{P_3(s,l)\})\} & \textit{otherwise} \end{cases}$$

$$\delta(s \vdash EX\ \varphi) = \sqcup^{+} \Big(\bigsqcap_{s' \in \overrightarrow{S}(s)} \delta(s' \vdash \varphi) \sqcap \{M(\{P_2(s,s')\})\} \Big)$$

$$\delta(s \vdash AX\ \varphi) = \sqcup^{+} \Big(\bigsqcup_{s' \in \overrightarrow{S}(s)} \delta(s' \vdash \varphi) \sqcup \{M(\{P_1(s,s')\})\} \Big)$$

$$\delta(\varphi_1 \vee \varphi_2) = \sqcup^{+} \big(\delta(s \vdash \varphi_1) \sqcup \delta(s \vdash \varphi_2)\big)$$

$$\delta(\varphi_1 \wedge \varphi_2) = \sqcup^{+} \big(\delta(s \vdash \varphi_1) \sqcap \delta(s \vdash \varphi_2)\big)$$

The configurations that follow from the rules (5) up to (8) have only one child configuration, thus the value of the function δ in these configurations are equivalent to the value defined in their single child. Hence, if $s \vdash \varphi$ is a configuration defined from one of the rules between (5) and (8) and $s \vdash \psi$ is the single configuration reached from it, then $\delta(s \vdash \varphi) = \delta(s \vdash \psi)$.

Definition 21. *Let M be a KMTS, s and s' states of M, and φ a CTL formula. The contraction model checking is a colouring function χ defined as follows:*

$$\chi(s \vdash \varphi) = \begin{cases} \top & \textit{iff } \delta(s \vdash \varphi) = \{M\} \\ F & \textit{iff } \delta(s \vdash \varphi) = \emptyset \\ \bot & \textit{otherwise} \end{cases}$$

6 Conclusions

In this work we developed a model checking game approach for model checking a KMTS interpreted as a set of CTL models. The works [11] and [5] interpret a KMTS as an abstraction of a concrete model to deal with the explosion state problem in CTL and μ–calculus model checking and define a 3-valued model checking which we take as reference for our contraction model checking. The works [1] and [2] also address the abstraction problem through partial Kripke structures which has indeterminations only in their states. The work reported in [12] considers the abstraction approach to deal with partial system specification through partial Kripke structures w.r.t Linear Tree Logic. In [8] the authors extend this structure with transitions that represent possibilities defining the KMTS structures. In [4] and [7] the authors consider MTS (Modal Transitions Systems) to deal with the abstraction approach. In these works a Kleene 3-valued logic or an equivalent interpretation is applied, while for the KMTS interpretation as a set of CTL models it does not hold. Moreover, none of these approaches

interprets a KMTS as a set of CTL models and to the best of our knowledge no other work considers this interpretation.

We argue that in order to verify a set of Kripke structures, model checking a KMTS that represents this set is on average better than model checking each CTL model at a time in the set. Since the determinations of a KMTS M are present in all the CTL models M represents, we can determine the truth value of some property common to all CTL models at once over M and we agree it can lead to a polynomial algorithm on average case. However, in the worst case since a KMTS represents an exponential set of CTL models, model checking a KMTS as a set of Kripke structures is NP-complete as we have already proven. Despite this, the presentation of this proof is beyond the scope of this work. We are investigating polynomial algorithms on average case and we intend to conclude this investigation providing efficient algorithms for model checking a KMTS as a set of CTL models in the future.

References

1. Bruns, G., Godefroid, P.: Model checking partial state spaces with 3-valued temporal logics. In: Halbwachs, N., Peled, D. (eds.) CAV'99, LNCS, vol. 1633, pp. 274–287. Springer, Heidelberg (1999)
2. Bruns, G., Godefroid, P.: Generalized model checking: Reasoning about partial state spaces. In: Palamidessi, C. (ed.) CONCUR 2000, LNCS, vol. 1877, pp. 168–182. Springer, Heidelberg (2000)
3. Clarke, E.M., Grumberg, O., Peled, D.A.: Model checking. MIT press (1999)
4. Godefroid, P., Huth, M., Jagadeesan, R.: Abstraction-based model checking using modal transition systems. In: Larsen, K., Nielsen, M. (eds.) CONCUR 2001, LNCS, vol. 2154, pp. 426–440. Springer, Heidelberg (2001)
5. Grumberg, O., Lange, M., Leucker, M., Shoham, S.: When not losing is better than winning: Abstraction and refinement for the full μ-calculus. Information and Computation 205(8), 1130 – 1148 (2007)
6. Guerra, P., Andrade, A., Wassermann, R.: Toward the revision of CTL models through kripke modal transition systems. In: Iyoda, J., de Moura, L. (eds.) SBMF 2013, LNCS, vol. 8195, pp. 115–130. Springer, Heidelberg (2013)
7. Huth, M.: Model checking modal transition systems using kripke structures. In: Cortesi, A. (ed.) VMCAI 2002, LNCS, vol. 2294, pp. 302–316. Springer, Heidelberg (2002)
8. Huth, M., Jagadeesan, R., Schmidt, D.: Modal transition systems: A foundation for three-valued program analysis. In: Sands, D. (ed.) ESOP 2001, LNCS, vol. 2028, pp. 155–169. Springer, Heidelberg (2001)
9. Huth, M., Ryan, M.: Logic in Computer Science: Modelling and reasoning about systems. Cambridge University Press (2004)
10. Kleene, S.C., de Bruijn, N., de Groot, J., Zaanen, A.C.: Introduction to metamathematics. Van Nostrand New York (1952)
11. Shoham, S., Grumberg, O.: A game-based framework for CTL counterexamples and 3-valued abstraction-refinement. ACM Trans. Comput. Logic 9(1) (2007)
12. Wehrheim, H.: Bounded model checking for partial kripke structures. In: Fitzgerald, J., Haxthausen, A., Yenigun, H. (eds.) ICTAC 2008, LNCS, vol. 5160, pp. 380–394. Springer, Heidelberg (2008)