SBMC : Symmetric Bounded Model Checking

Brahim NASRAOUI
LIP2 and Faculty of
Sciences of Tunis
Campus Universitaire 2092 -
El Manar Tunis
Tunisia
brahim.nasraoui@gmail.com

Syrine AYADI
LIP2 and Faculty of
Sciences of Tunis
Campus Universitaire 2092 -
El Manar Tunis
Tunisia
syrine.ayadi@ensi.rnu.tn

Riadh ROBBANA
LIP2 and Polytechnic
School of Tunisia
B.P. 743 - 2078 La Marsa
Fax : +21671748843
Tunisia
riadh.robana@fst.rnu.tn

This paper deals with systems verification techniques, using Bounded Model Checking (BMC). We present a new approach that combines BMC with symmetry reduction techniques. Our goal is to reduce the number of transition sequences, which can be handled by a SAT solver, used in the resolution of verification problems. In this paper, we generate a reduced model by exploiting the symmetry of the original model, which contains only transition sequences that represent the equivalence classes of the symmetric transition sequences. We consider the construction of a new Boolean formula that manipulates only representative transition sequences. In our technique, we present a method that combines the symmetry reduction technique with BMC for the reduction of the space and time of Model Checking.

Model Checking; Symmetry reduction; SAT; Boolean Formula; Bounded Model Checking; Formal methods.

1. INTRODUCTION

The main challenge behind model checking is the state explosion problem. Kupferman et al (2000) describe how the classic methods are unable to check properties on large systems in a reasonable time. Historically, several methods were developed to solve this problem, one can for example change the structure of data being used to encode the system. Thus, in addition to the automata, one can use OBDDs, as Coudert et al (1990); Bruch et al (1992); de Alfaro et al (2000), or encoding in terms of SAT clauses, as McMillan (2003), while Bounded Model Checking techniques unroll the model for a fixed number of steps k , check whether a property violation can occur in k or fewer steps, and encode the restricted model as an instance of SAT. The process can be repeated until all possible violations have been ruled out. In the other hand, symmetry reduction methods exploit symmetry in order to efficiently verify its temporal property.

Model checking, as defined in Clarke et al (1999), is the most important technique for verifying systems. The use of BDDs and SAT in symbolic model checking, in McMillan (1993), has led to the success of this technique in the verification of many system designs. However, explicit and BDD-based model checking suffer from the state space explosion problem. In order to solve this, symmetry reduction techniques have been used in

symbolic model checking: Emerson et al (1996); Clarke et al (1996, 1998); Jha (1996); Barner et al (2002); Emerson et al (2003). In the symmetric system, two states are considered equivalents if they have the same behavior. Many works have applied symmetry based reduction methods for model checking concurrent systems, Vardi (1996); Clarke et al (1993). This method has been shown to be an effective technique in both explicit and symbolic model checking, which exploits the fact that many systems are composed by interchangeable components, and therefore it may be sufficient to consider a smaller version of the symmetrical state space, called the reduced model.

The basic idea behind the reduction of symmetry is to partition the state space into equivalence classes and to choose one or more representatives from each equivalence class in the model during model checking. Previous studies have shown reductions in both memory and time consumption when exploiting symmetries in model checking.

Symmetry reduction in explicit model checking reduce the state space in the initial model as in Emerson et al (1996). Many works have considered the combination of symmetry reduction with symbolic model checking based on BDD, in Clarke et al (1996); Barner et al (2002); Emerson et al (2003). They construct an orbit relation to generate the reduced model and they choose a unique representative for each orbit. The computation of

the orbit relation can be done in polynomial time for certain practical symmetric systems. However, it consumes exponential time in general, Clarke et al (1998). Clarke et al (1996, 1998) have also proposed to allow several representatives for each orbit. On the fly representatives have also been proposed in Barner et al (2002), where at each iteration the fixed point calculation, states whose symmetric states are not encountered in the previous iterations are chosen to be the representative of their respective orbits. Thus, it is possible to have several representatives for each orbit. Another way of exploiting symmetry is to translate the description of the symmetric system in the generic form, where the local state variables of symmetrical components are substituted by global counter variables, and then translate the generic representation into corresponding BDD, as in Emerson et al (2003). These translations require modifications to the front-end verification tool which is done obviously.

In this paper, we propose a method of symmetry reduction in Bounded Model Checking. First of all by reducing the number of the sequences of the checked model, when adding some clauses which inhibit the effect of the nonrepresentative transitions of their classes of equivalence in order to generate a Boolean formula which represents the system, we also develop an algorithm (Symmetric Bounded Model Checking) which makes possible to check complex symmetrical systems. In this work we will combine the technique of reduction by the method of symmetry and BMC. Our work consists in generating a Boolean formula that holds account only symmetrical sequences of transitions, without the construction of the small-scale model. The model checking problem amounts to solve the satisfiability of this formula.

The structure of this paper is as follows: In section 2, we give preliminary definitions. The principle of bounded model checking is detailed in section 3. The section 4 presents the notions of representative states and transitions of a symmetric model, defined with permutation function and equivalence classes. Our contribution, namely the symmetric Bounded model checking is detailed in the section 5, while section 6 gives a proof of its correction. An illustrative example is given at the section 7. Finally, section 8 concludes and outlines future work.

2. PRELIMINARY DEFINITIONS

2.1. Kripke structures

A Kripke structure is a type of finite state machine, used to represent the behaviour of a system in the Model Checking. It is a graph whose nodes

represent the reachable states of the system and whose edges represent state transitions. A labeling function maps each node to a set of properties that hold in the corresponding state. We use transitions system to represent all the possible executions of a given system. Formally, a transition system defined by a Kripke structure as follows:

Definition A Kripke structure constructed over a finite set of atomic propositions: $AP = \{P_1, P_2, \dots, P_n\}$ is defined by $M = (S, I, R, L)$ where:

S is a set of states,

$I \subseteq S$: set of initial states

$L : S \rightarrow 2^{AP}$ labelling function which labels each state with atomic propositions that are true in S , and

$R \subseteq S \times S$ is the transition relation.

2.2. LTL : Linear Temporal logic

To specify properties, we use LTL : linear temporal logic in Pnueli (1977); Manna et al (1991). An LTL formula ϕ is defined over a set of atomic propositions AP , and has the following syntax :

1. $\psi \in AP$ is an LTL formula.
2. If ψ and φ are LTL formulae then so are $\neg\psi$, $X\psi$, $\psi U \varphi$, $\psi R \varphi$, $\psi \wedge \varphi$ and $\psi \vee \varphi$.

The operators are the next-time operator X , the until operator U , and its dual the release operator R . Each formula defines a set of infinite words (models) over 2^{AP} . Let $\pi \in (2^{AP})^\omega$ be an infinite word. We denote the suffix of a word $\pi = \sigma_0\sigma_1\sigma_2\dots$ by $\pi^i = \sigma_i\sigma_{i+1}\sigma_{i+2}\dots$ where $\sigma_i \in 2^{AP}$, and π_i denotes the prefix $\pi_i = \sigma_0\sigma_1\dots\sigma_i$. When a formula ψ defines a word π at time i this is denoted $\pi^i \models \psi$. The set of infinite words defined by a formula ψ is $\{\pi \in (2^{AP})^\omega \mid \pi \models \psi\}$.

3. BOUNDED MODEL CHECKING BMC

The success of SAT solvers, described in McMillan (2002, 2003) in Boolean Formula resolution has contributed to the appearance of BMC, in Biere et al (2001); Biere et al (1999); McMillan (2002, 2003); Shtrichman et al (2000), since BMC uses SAT solvers in the resolution of the Model Checking problem.

The basic idea behind BMC is to consider only finite prefixes of paths which can be used as witnesses of the Model Checking problem and in which the path is restricted to a certain bound k . The bound is incremented until a witness is found. Two types of paths are considered: paths with loops (see figure 1.b), and paths without loops (see figure 1.a). Loop paths are interpreted as infinite paths (which contain a transition from the last state to a previous state), this path admits a finite length. If the path does not have a loop, this implies that the prefix can not

Figure 1: Path with and without loop

decide the behavior of the path after the k^{th} state, it implies that witnesses are formed from the paths with loops since these paths have a finite length.

Let M be a Kripke structure and f be an LTL formula, we adopt the following notations:

- $\pi(k)$ is the k^{th} state.
- $[[M]]$ is the evaluation of f in a sequence π , where f is an LTL formula.

Definition For $l \leq k$, we call a path π a (k, l) -loop if $R(\pi(k), \pi(l))$ and $\pi = u.v^w$ with $u = (\pi(0), \dots, \pi(l-1))$ and $v = (\pi(l), \dots, \pi(k))$. We say that π is a k -loop, if there is $k, k \geq l \geq 0$ for which π is a (k, l) -loop.

There are in the BMC approaches two semantics: a semantic defined in the loop paths and a semantic defined in a path without a loop. Bounded semantics manipulate only prefixes of paths which have a bounded length k .

Definition (Bounded semantics for a path with a loop) Let $k \geq 0$ and π be a k -loop. Thus an LTL formula f is valid in a sequence π with bound k (denoted by $\pi \models_k f$) iff $\pi \models f$.

Definition (Bounded semantics for a path without a loop) Let $k \geq 0$, and π be a path which is not a k -loop. An LTL formula f is valid along π with bound k , (denoted by $\pi \models_k f$) iff $\pi \models_k^0 f$ where

$\pi \models_k^i p$	iff	$p \in L(\pi(i))$
$\pi \models_k^i \neg p$	iff	$p \notin L(\pi(i))$
$\pi \models_k^i f \wedge g$	iff	$\pi \models_k^i f$ and $\pi \models_k^i g$
$\pi \models_k^i f \vee g$	iff	$\pi \models_k^i f$ or $\pi \models_k^i g$
$\pi \models_k^i \mathbf{G}f$	is	always false
$\pi \models_k^i \mathbf{F}f$	iff	$\exists j, i \leq j \leq k. \pi \models_k^j f$
$\pi \models_k^i \mathbf{X}f$	iff	$i < k$ and $\pi \models_k^{i+1} f$
$\pi \models_k^i f \mathbf{U} g$	iff	$\exists j, i \leq j \leq k. \pi \models_k^j g$ and $\forall n, i \leq n < j. \pi \models_k^n f$
$\pi \models_k^i f \mathbf{R} g$	iff	$\exists j, i \leq j \leq k. \pi \models_k^j f$ and $\forall n, i \leq n < j. \pi \models_k^n g$

If π is not a k -loop, then $\mathbf{G}f$ is not valid along π , in the bounded semantics with a bound k , because f may not satisfy the $(k+1)^{\text{th}}$ state of π . This induces

that the duality between $\mathbf{G}$ and $\mathbf{F}$ does not hold in the bounded semantics.

4. REPRESENTATIVE STATES AND TRANSITIONS

In the symmetry approach, automorphisms of the global model are exploited. Given a property ϕ specified in a temporal logic, and a model. Symmetry reduction methods consist of the generation of a model which considers only the representatives of equivalence classes, this model is named a quotient structure, and checks the formula ϕ on the model using traditional model checking algorithms.

In the rest of this paper, we will define the symmetry group C induced by a Kripke structure M , we introduce the notion of symmetric transition sequences and we give the definition of its representative. The following definitions are useful in the introduction of the process of our method.

Definition [Rintanen (2003)]

For a Kripke structure $M = (S, I, R, L)$, a symmetry group C defined over M is a pair $\langle \sigma, \tau \rangle$ such that:

- $\sigma : S \rightarrow S$ is a permutation function defined over S ,
- $\tau : R \rightarrow R$ is a permutation function defined over R ,
- $\forall (s_1, s_2) \in S \times S$ and $t \in R, (\sigma(s_1), \sigma(s_2)) = \tau(t)$ iff $(s_1, s_2) = t$ and
- $L(s) = L(\sigma(s)), \forall s \in S$.

In the following we define the notion of state sequences and transition sequences in a kripke model M .

Definition Let $M = (S, I, R, L)$ be a kripke structure, π and π' two finite state sequences in the model M such that:

$$\forall i, 0 \leq i \leq k, s_i \in S, \pi = s_0 \dots s_k \text{ and } \pi' = s'_0 \dots s'_k.$$

We say that π and π' are two symmetric state sequences if and only if:

$$\forall i, 0 \leq i \leq k, s'_i = \sigma(s_i), (s_i, s_{i+1}) \in R \text{ and } (s'_i, s'_{i+1}) \in R.$$

where σ is a permutation function over states.

Definition Let $M = (S, I, R, L)$ be a kripke structure, π and π' are two finite transition sequences in the model M such that:

$$\forall i, 1 \leq i \leq k, t_i \in R, t'_i \in R, \pi = t_1 \dots t_k \text{ and } \pi' = t'_1 \dots t'_k.$$

We say that: π and π' are two symmetric transition sequences iff:

$$\forall i, 1 \leq i \leq k, t_i = \tau(t'_i).$$

where τ is a permutation function over transitions.

Definition A transition equivalence class of t_i denoted by Cl_i is a set of transitions that verifies:

$$\forall t \in Cl_i, t \in R \text{ and } t = \tau(t_i).$$

We denote by $\underline{R}$ the set of representative transitions. These transitions represent the equivalence classes belonging to it, $\underline{R} \subseteq R$. Similarly, we denote by $\underline{S}$ the set of representative states. These states are the representatives of their equivalence classes, $\underline{S} \subseteq S$. And consequently $\underline{I}$ is the set of initial representative states. These states are the representatives of their equivalence classes, $\underline{I} \subseteq I$. Then $\underline{M} = (\underline{S}, \underline{I}, \underline{R}, \underline{L})$ is the representative model of $M = (S, I, R, L)$. We note that each representative transition must have a common state with the previous one.

Property 4.1 A representative transition denoted by $\underline{t}$ is a transition that represents its equivalence class and verifies: $\forall i, 2 \leq i \leq k, t_i \in Cl_i, \exists s_{i-1}, s_i, s_{i+1} \in S, t_i = (s_i, s_{i+1})$ and $t_{i-1} = (s_{i-1}, s_i)$

Remark For $i = 1$ the transition $\underline{t}_1$ is a transition that must begin from an initial state $\underline{t}_1 = (s, s')/s \in \underline{I}$ and $s' \in S$.

In the following we give some definitions that are used in the proof of the equivalence between the two Boolean formula. We will introduce the notion of representative path and non-representative path.

Definition Let $M = (S, I, R, L)$ be a kripke structure, and π a finite path in the model M such that:

$$\forall i, 1 \leq i \leq k, t_i \in R, \pi = t_1 \dots t_k.$$

We say that π is a representative transition sequence denoted by π_{rep} if $\forall i, 1 \leq i \leq k, t_i \in \underline{R}$ and if $t_i = (s_i, s_{i+1}), t_{i+1} = (s'_i, s'_{i+1}) \implies (s_{i+1}, s'_i) \in R$ (s.t. t_i is the representative of the equivalence classes $\underline{R}$).

Definition Let $M = (S, I, R, L)$ be a kripke structure, and π a finite path in the model M such that:

$$\forall i, 1 \leq i \leq k, t_i \in R, \pi = t_1 \dots t_k.$$

We say that π is a non representative transition sequence denoted by π_{nrep} if $\exists i, 1 \leq i \leq k, t_i \notin \underline{R}$ (s.t. t_i is not the representative of the equivalence classes $\underline{R}$).

We define a Boolean function that tests if a transition belongs to the set of representative transitions

which detects the non representative transitions and eliminates these sequences of transitions in the new Boolean formula F' .

In the following we will give the formal definition of this function:

Definition Let R be the set of the transition relation, We define T like a function that is defined over R as follows $T(t) = 1$ iff $t \in \underline{R}$.

$$T : R \rightarrow \{0, 1\} / \forall t \in R, T(t) = \text{true} \text{ iff } t \in \underline{R}.$$

5. SYMMETRY-BMC

5.1. Selecting reduced model

In this section we present our algorithm *Representative_Model*, this algorithm solves two problems induced from the symmetry reduction technique:

- 1 the construction of representative states and transition sets.
- 2 the choice of the representatives of these states and transitions.

Algorithm: *Representative_Model*

Input: M : initial model

Output: $\underline{R}$: representative transitions

Begin

Initially: $\underline{R} = \emptyset$

1. compute the set of the equivalence classes $CA = \bigcup_i Cl_i$ on the transitions of M .
2. for all $Cl_i \in CA$ with $i \leq k$ do
3. select one representative $\underline{t}_i$ of Cl_i
4. $\underline{R} := \underline{R} \cup \{\underline{t}_i\}$

end

The different steps presented in our algorithm *Representative_Model* are as follows:

- Compute transition equivalence classes of M . In this step, symmetric transition sequences are obtained from their orbits (equivalence classes).
- With the condition $i \leq k$, the number of transitions in a path is limited by the bound k . Thus, we are interested only in the equivalence classes that are reachable in k iterations.
- Select one representative transition from each class of Cl_i . During this step the algorithm

select one representative transition from Cl_i verifying the property 4.1.

- Compute $\underline{R} := \underline{R} \cup \{\underline{t}\}$
This step allows the construction of the set $\underline{R}$ that represents the set of transitions which in turn represents the orbit of the transitions. $\underline{R}$ initially contains $\emptyset$.

This algorithm constructs a set of transitions that represent the reduced model by symmetry.

This set is denoted by $\underline{R}$. We will work on this set in the generation of the representative transition sequences, for this reason we will generate a new Boolean formula which handles only representative transition sequences:

It begins from the set of representative transitions which represent the transitions starting from the initial states, and in each iteration it selects one representative transition $\underline{t}$. Model Checking with SAT instances will speed up the process of resolution, this is due to the reduction of the transition sequences.

5.2. Transformation of the BMC to the SBMC

In this section we focus on the translation from the problem of BMC to the SBMC. The initial system is then directly modeled by the new Boolean formula F' which does not handle all the transition sequences but only the representative of all the symmetric transition sequences. This translation will speedup the process of searching for counter examples and scales better than the classic approach of Bounded Model Checking.

Given a Kripke structure M , an LTL formula f and a bound k , we construct propositionnal formula F' that models only representative transition sequences. Let $\pi_{rep} = \underline{t}_1 \dots \underline{t}_k$ be a finite representative transition sequence that forms a representative path.

Each transition is represented by a binary codage over a set of variables. The formula F' takes into account these transitions and searches for an encodage such that F' is satisfiable if and only if π_{rep} is a valid sequence in the model M that satisfies f . This formula F' constructs a path, that is a sequence of states representing sequences of transitions that belong to the representatives of the equivalence classes.

Finally this formula generates the representatives of all symmetric paths satisfying f . The construction of the paths begins from the initial states, and is followed by the research of a valid sequence of the representative transitions that satisfy the LTL formula f to be checked. In this formula F' , the state s_0 must be an initial state and the transitions (s_i, s_{i+1}) must be in R and must be a representative transition (i.e. (s_i, s_{i+1}) is in $\underline{R}$).

$$F' = I(s_0) \wedge \bigwedge_{i=0}^k (R(s_i, s_{i+1}) \wedge T(s_i, s_{i+1})) \wedge$$

$$\left((\neg L_k \wedge [[f]]_k^0) \vee \bigvee_{l=0}^k (\iota L_k \wedge \iota [[f]]_k^0) \right).$$

In the following we prove that the resolution of the BMC problem can be replaced by our method when the system exhibits structural symmetry in its specification. This technique is called SBMC.

6. EQUIVALENCE BETWEEN FORMULAS

In the previous section we introduced a new technique called SBMC. This technique is based on BMC and exploits the symmetry of the problem, for generating a new formula F' that handles only transition sequences representing the part of the initial model or the reduced model.

The Boolean formula F' adds new clauses that inhibit the fact of symmetric transitions of the initial model. In this following, we establish the equivalence relation between the two Boolean Formulas F and F' .

Proposition 6.1 *For a Kripke structure $M = (S, I, R, L)$, we have: the formula F that can be generated from the initial model is equivalent to the the formula F' that represents the reduced model $\underline{M}$:*

$$F \iff F'$$

In the sequel, we prove the equivalence between F and F' , which will be proceeded by the two steps:

1. $F' \Rightarrow F$,
2. $F' \Leftarrow F$.

Proof Let $M = (S, I, R, L)$ be a kripke structure, f an LTL formula and k an integer, such that: $k \geq 0$.

Let us prove that $F \iff F'$.

Since the formulas $F \iff F'$ and $F' \Rightarrow F \wedge F' \Leftarrow F$ are equivalent, we will prove the validity of $(F' \Rightarrow F) \wedge (F' \Leftarrow F)$ in two steps.

First of all, let us prove the first implication, namely $F' \Rightarrow F$.

We have: $[[M]]_k := I(s_0) \wedge \bigwedge_{i=0}^k R(s_i, s_{i+1})$, and $F := [[M, f]]_k := [[M]]_k \wedge \left((\neg L_k \wedge [[f]]_k^0) \vee \bigvee_{l=0}^k (\iota L_k \wedge \iota [[f]]_k^0) \right)$,

By substituting $[[M]]_k$ value within F expression, we obtain:

$$F = I(s_0) \wedge \bigwedge_{i=0}^k R(s_i, s_{i+1}) \wedge \left((\neg L_k \wedge [[f]]_k^0) \vee \bigvee_{l=0}^k (\iota L_k \wedge \iota [[f]]_k^0) \right), \text{ and we have either: } F' =$$

$I(s_0) \wedge \bigwedge_{i=0}^k (R(s_i, s_{i+1}) \wedge T(s_i, s_{i+1})) \wedge \left((\neg L_k \wedge [[f]]_k^0) \vee \bigvee_{l=0}^k (L_k \wedge l[[f]]_k^0) \right)$. The demonstration of the implication $F' \Rightarrow F$, is as follows:
While replacing F and F', with their respective expressions, we obtain the following statement:

$$I(s_0) \wedge \bigwedge_{i=0}^k (R(s_i, s_{i+1}) \wedge T(s_i, s_{i+1})) \wedge \left((\neg L_k \wedge [[f]]_k^0) \vee \bigvee_{l=0}^k (L_k \wedge l[[f]]_k^0) \right) \Rightarrow I(s_0) \wedge \bigwedge_{i=0}^k R(s_i, s_{i+1}) \wedge \left((\neg L_k \wedge [[f]]_k^0) \vee \bigvee_{l=0}^k (L_k \wedge l[[f]]_k^0) \right) \equiv$$

And since $(A \Rightarrow B \equiv \neg A \vee B)$, the previous expression will be transformed as follows:

$$\neg \left(I(s_0) \wedge \bigwedge_{i=0}^k (R(s_i, s_{i+1}) \wedge T(s_i, s_{i+1})) \wedge \left((\neg L_k \wedge [[f]]_k^0) \vee \bigvee_{l=0}^k (L_k \wedge l[[f]]_k^0) \right) \right) \vee \left(I(s_0) \wedge \bigwedge_{i=0}^k R(s_i, s_{i+1}) \wedge \left((\neg L_k \wedge [[f]]_k^0) \vee \bigvee_{l=0}^k (L_k \wedge l[[f]]_k^0) \right) \right) \equiv$$

By applying De Morgan's law, namely $(\neg(A \wedge \dots \wedge B) \equiv (\neg A \vee \dots \vee \neg B))$, on the previous statement, the later will be written as the follower:

$$\neg I(s_0) \vee \neg \left(\bigwedge_{i=0}^k (R(s_i, s_{i+1}) \wedge T(s_i, s_{i+1})) \right) \vee \neg \left((\neg L_k \wedge [[f]]_k^0) \vee \bigvee_{l=0}^k (L_k \wedge l[[f]]_k^0) \right) \vee \left(I(s_0) \wedge \bigwedge_{i=0}^k R(s_i, s_{i+1}) \wedge \left((\neg L_k \wedge [[f]]_k^0) \vee \bigvee_{l=0}^k (L_k \wedge l[[f]]_k^0) \right) \right) \equiv$$

Now, let us consider the following equivalence:
 $a \vee (\neg a \wedge b) \equiv a \vee b$.

For $a := \neg I(s_0)$,

$$b := \left(I(s_0) \wedge \bigwedge_{i=0}^k R(s_i, s_{i+1}) \wedge \left((\neg L_k \wedge [[f]]_k^0) \vee \bigvee_{l=0}^k (L_k \wedge l[[f]]_k^0) \right) \right).$$

We will obtain the following statement: $\neg I(s_0) \vee \neg \left(\bigwedge_{i=0}^k (R(s_i, s_{i+1}) \wedge T(s_i, s_{i+1})) \right) \vee \neg \left((\neg L_k \wedge [[f]]_k^0) \vee \bigvee_{l=0}^k (L_k \wedge l[[f]]_k^0) \right) \vee \left(\bigwedge_{i=0}^k R(s_i, s_{i+1}) \wedge \left((\neg L_k \wedge [[f]]_k^0) \vee \bigvee_{l=0}^k (L_k \wedge l[[f]]_k^0) \right) \right) \equiv$

If we consider another time the previous equivalence $a \vee (\neg a \wedge b) \equiv a \vee b$, and while substituting

$$a := \neg \left((\neg L_k \wedge [[f]]_k^0) \vee \bigvee_{l=0}^k (L_k \wedge l[[f]]_k^0) \right), \text{ and}$$

$$b := \left((\neg L_k \wedge [[f]]_k^0) \vee \bigvee_{l=0}^k (L_k \wedge l[[f]]_k^0) \right), \text{ we get the}$$

following statement: $\neg I(s_0) \vee \neg \left(\bigwedge_{i=0}^k (R(s_i, s_{i+1}) \wedge T(s_i, s_{i+1})) \right) \vee \bigwedge_{i=0}^k R(s_i, s_{i+1}) \vee \neg \left((\neg L_k \wedge [[f]]_k^0) \vee \bigvee_{l=0}^k (L_k \wedge l[[f]]_k^0) \right) \equiv$

The transformations between the following formulas are ensured by De Morgan's laws as the sequel:

$$\neg I(s_0) \vee \bigvee_{i=0}^k \neg (R(s_i, s_{i+1}) \wedge T(s_i, s_{i+1})) \vee \bigwedge_{i=0}^k R(s_i, s_{i+1}) \vee \neg \left((\neg L_k \wedge [[f]]_k^0) \vee \bigvee_{l=0}^k (L_k \wedge l[[f]]_k^0) \right) \equiv$$

$$I(s_0) \vee \bigvee_{i=0}^k (\neg R(s_i, s_{i+1}) \vee \neg T(s_i, s_{i+1})) \vee \bigwedge_{i=0}^k R(s_i, s_{i+1}) \vee \neg \left((\neg L_k \wedge [[f]]_k^0) \vee \bigvee_{l=0}^k (L_k \wedge l[[f]]_k^0) \right) \equiv \neg I(s_0) \vee \bigvee_{i=0}^k \neg R(s_i, s_{i+1}) \vee \bigvee_{i=0}^k \neg T(s_i, s_{i+1}) \vee \bigwedge_{i=0}^k R(s_i, s_{i+1}) \vee \neg \left((\neg L_k \wedge [[f]]_k^0) \vee \bigvee_{l=0}^k (L_k \wedge l[[f]]_k^0) \right) \equiv \neg I(s_0) \vee \neg \bigwedge_{i=0}^k R(s_i, s_{i+1}) \vee \bigvee_{i=0}^k \neg T(s_i, s_{i+1}) \vee \bigwedge_{i=0}^k R(s_i, s_{i+1}) \vee \neg \left((\neg L_k \wedge [[f]]_k^0) \vee \bigvee_{l=0}^k (L_k \wedge l[[f]]_k^0) \right) \equiv true.$$

because $\bigwedge_{i=0}^k R(s_i, s_{i+1}) \vee \neg \bigwedge_{i=0}^k R(s_i, s_{i+1}) \equiv true$.

Concerning the second part of the proof $F \Rightarrow F'$:
We have $\forall \pi \in M, F(\pi) \Rightarrow F'(\pi)$, which means that if we have $F(\pi)$ true, $F'(\pi)$ will be either true.

In the initial model we have two kind of paths: representative path π_{rep} and non-representative path π_{nrep} , this implies that the path $\pi \in \{\pi_{rep}, \pi_{nrep}\}$.

- if $\pi = \pi_{rep}$ then $F(\pi) = F'(\pi)$, because π is a representative transition sequence both in the initial model and in the reduced model by symmetry.
- otherwise ($\pi = \pi_{nrep}$) in this case $\pi_{rep} = \sigma(\pi) \Rightarrow F(\pi_{rep}) = F(\pi)$ or $F(\pi_{rep}) = F'(\pi_{rep}) \Rightarrow F(\pi) = F'(\pi_{rep}) \Rightarrow F(\pi) = F'(\sigma(\pi))$

Proposition 6.2 Let $M = (S, I, R, L)$ be a kripke structure, f an LTL formula, and suppose that $\underline{M}$ represents the reduced model of M , we have:

$$M \models f \iff \underline{M} \models f$$

Proof Let F be a Boolean formula which represents the initial model, and F' the Boolean formula that represents the reduced model $\underline{M}$.

$$M \models f \iff \exists k, [[M, f]]_k \text{ is satisfiable}$$

F is satisfiable, due to the fact that $F \Leftrightarrow F'$, this implies that F' is similar to F , which means that they have the same behaviors.

F' models the symmetric model or the reduced model, then we have:

$$\underline{M} \models f \iff \exists k, F' \text{ is satisfiable}$$

Theorem 6.3 Let $M = (S, I, R, L)$ be a kripke structure, f an LTL formula. $M \models f$ iff there exists $k, k \geq 0$ such that: $M \models_k f$.

The proof of this theorem can be founded in, Biere et al (2003), and from it we can derive the following Corollary which specify the fact that if an initial model

satisfies an LTL formula f then the reduced model by symmetry do either.

Corollary 6.4 *Let $M = (S, I, R, L)$ be a kripke structure, f an LTL formula. $M \models f$ if there exists $k, k \geq 0$ such that: $\underline{M} \models_k f$.*

Proof The result is trivial due to Theorem 6.3.

7. EXAMPLE

In this example we don't construct the transition system, symmetries are derived from an implicit representation of the model and the transition system is given in figure 2 only to illustrate our approach.

Consider the transition system with symmetries shown in figure 2.

This structure has the following automorphisms:

Figure 2: Reduced Model $\underline{M}$

$Cl_1 = \{1\}$, $Cl_2 = \{2, 3\}$, $Cl_3 = \{4\}$, $Cl_4 = \{5, 6\}$, and $Cl_5 = \{7\}$. Therefore these automorphisms induces the transition equivalence classes: $\langle(1, 2), (1, 3)\rangle$, $\langle(2, 4), (3, 4)\rangle$, $\langle(1, 7), \langle(7, 5), (7, 6)\rangle, \langle(5, 4), (6, 4)\rangle$ and $\langle(5, 2), (6, 3)\rangle$. Thus, paths that contain only representative transition sequences in their transitions are named the representative paths.

Therefore, the following representative paths induced from the initial model and represented by the Boolean formula are:

$\pi'_1 = 1, 2, 4$, $\pi'_2 = 1, 7, 5, 4$, and $\pi'_3 = 1, 7, 5, 2, 4$.

However, the standard approach of BMC is to consider all paths in the transition system which would be:

$\pi_1 = 1, 2, 4$, $\pi_2 = 1, 7, 5, 4$, $\pi_3 = 1, 3, 4$, $\pi_4 = 1, 7, 6, 4$, $\pi_5 = 1, 7, 5, 2, 4$, and $\pi_6 = 1, 7, 6, 3, 4$.

8. CONCLUSION

We have presented in this work a new method, which combines the reduction technique with BMC. This approach consists of the representation of Model Checking problems with a Boolean formula. This formula generates a set of representative

transitions, these transition sequences are formed by sequences of representative transitions representing their equivalence classes. Thus, the number of sequences handled, while the model checking of the model, is restricted to representative transition sequences, which have much smaller than the total number of all the transition sequences in the initial model. Therefore, the use of our approach speeds up the process of model checking, this is due to the non treatment of the non representative transition sequences, and to the use of the SAT solvers in the resolution of the problem, which is the search of counter examples of the property to be checked. Thus, we can find a counter example faster than the other techniques.

On the one hand, our approach is characterized by the fact that it generates equivalence classes, and choose a representative of these classes. Our method generates a formula which models only representative transitions, without the construction of the reduced model. The problem of the model checking is reduced to the satisfiability of the Boolean formula.

As future work we plan to consider the following points:

- Extend our approach for timed systems.
- Determine the best bound k by exploiting symmetry.
- Exploit the symmetry of the clauses generated in the Boolean formula before being solved by the SAT

9. REFERENCES

- Barner, S, Grumberg, O, *Combining symmetry reduction and under-approximation for symbolic model checking* In: International Conference on Computer-Aided Verification (CAV). (2002)
- Biere, A., Cimatti, A. Clarke, E.M., and Zhu, Y. *Symbolic model checking without BDDs* In the proceeding of the 5th International Conference on Tools and Algorithms for Construction and Analysis of Systems 193-207, 2001.
- Biere, A., Cimatti, A., Clarke, E.M., Fujita, M., Zhu, Y.: *Symbolic model checking using SAT procedures instead of BDDs* In: Design Automation Conference (DAC), 1999.
- Biere, A. Cimatti. A, Clarke, E. M., Strichman, O. and Zhu, Y. *Bounded model Checking*, In Highly Dependable Software, volume 58 of Advances in Computers. Academic Press, 2003.
- Bryant, R. E. *On the complexity of VLSI implementations and graph representations of Boolean functions with application to integer multiplication*

- IEEE Transactions on Computers, 40(2) :205-213, February 1991.
- Bryant, R. E. *Graph-based algorithms for Boolean function manipulation* IEEE Transactions on Computers, C-35 :677-691, 1986.
- Bryant, R. E. *Symbolic boolean manipulation with ordered binary decision diagrams* ACM Computing Surveys, 24(3) :293-318, September 1992.
- Burch, J. R., Clarke, E. M., McMillan, K. L., Dill, D. L. and Hwang, J. *Symbolic model checking : 1020 states and beyond* Information and Computation, 98(2) :142-170, 1992.
- Clarke, E. M., Filkorn, T., Jha, S. *Exploiting Symmetry In Temporal Logic Model Checking*, Proceedings of the 5th International Conference on Computer Aided Verification, p.450-462, June 28-July 01, 1993.
- Clarke, E.M., Enders, R., Filkorn, T., Jha, S, *Exploiting symmetry in temporal logic model checking* Formal Methods in System Design 9 (1996) 77104
- Clarke, E.M., Emerson, E.A., Jha, S., Sistla, A.P, *Symmetry reductions in model checking* In: International Conference on Computer Aided Verification (CAV). (1998)
- Clarke, E.M., Grumberg, O., Peled, D, *Model Checking* MIT Press (1999)
- Coudert, O. and Madre, J. C. *A unified framework for the formal verification of sequential circuits*, Proceeding of ICCAD, pages 126-129, 1990.
- de Alfaro, L., Kwiatkowska, M. Z., Norman, G., Parker, D. and Segala, R. *Symbolic model checking of probabilistic processes using MTBDDs and the kronecker representation* In Susanne Graf and Michael I. Schwartzbach, editors, TACAS, volume 1785 of Lecture Notes in Computer Science, pages 395-410. Springer, 2000.
- Emerson, E.A, Wahl, T, *On combining symmetry reduction and symbolic representation for efficient model checking*, In: Conference on Correct Hardware Design and Verification Methods (CHARME).
- Emerson, E.A., Sistla, A.P, *Symmetry and model checking* Formal Methods in System Design 9 (1996) 105131
- Ganai, M., Gupta, A., Ashar, P. *Efficient SAT-based unbounded symbolic model checking using circuit cofactoring* Proceeding of the International Conference on Computer-Aided Design (ICCAD), 2004.
- Jha, S. *Symmetry and Induction in Model Checking*. PhD thesis, School of Computer Science, Carnegie Mellon University (1996)
- KUPFERMAN, O., VARDI, M. Y., AND WOLPER, P *An automata-theoretic approach to. branching-time model checking JACM : Journal of the ACM*, 47, 2000.
- Manna, Z, and Pnueli, A. *The Temporal Logic of Reactive and Concurrent Systems: Specification*, Springer-Verlag, New York, 1991.
- McMillan, K.L, *Symbolic Model Checking: An Approach to the State Explosion Problem* Kluwer Academic Publishers (1993)
- McMillan, K.L. *Applying SAT methods in unbounded symbolic model checking* Proceeding of the International Conference on Computer-Aided Verification (CAV), 2002.
- McMillan, K.L. *Interpolation and SAT-based model checking* Proceeding of the International Conference on Computer-Aided Verification (CAV), 2003.
- Pnueli, A. *The Temporal Logic of Programs*, In Proceedings of the 18th IEEE Symposium on Foundations of Computer Science (FOCS 1977), pages 4657, 1977.
- Rintanen, J. *Symmetry Reduction for SAT Representations of Transition Systems* Proceedings of the Thirteenth International Conference on Automated Planning and Scheduling (ICAPS 2003), June 9-13, 2003, Trento, Italy.
- Shtrichman, O. *Tuning SAT checkers for bounded model checking* In proceeding of the 12th International Conference on Computer Aided Verification p.480-494, 2000.
- Vardi, M. Y.. *An automata-theoretic approach to linear temporal logic* In Logics for Concurrency, volume 1043 of LNCS, pages 238-266, 1996.