# A New Translation from ECTL* to SAT

Andrzej Zbrzezny

IMCS, Jan Długosz University in Częstochowa,
Al. Armii Krajowej 13/15, 42-200 Częstochowa, Poland.
a.zbrzezny@ajd.czest.pl

**Abstract.** In this paper we present a new translation from ECTL* to SAT and show that the proposed translation substantially increases the efficiency of verifying temporal properties using the Bounded Model Checking method. We have implemented our new translation and made experimental results, which demonstrate the efficiency of the method.

**keywords:** ECTL*, translation to SAT, Bounded Model Checking

## 1 Introduction

Bounded Model Checking (BMC) is a popular model checking technique for the verification of finite-state transition systems [2, 4]. BMC has been introduced as a technique complementary to symbolic BDD-based model checking. Given a finite transition system $\mathcal{M}$ and an ACTL* formula $\alpha$, the BMC procedure tries to find a natural number $k$ and a set of finite paths of length $k$ in $\mathcal{M}$, the so called $k$-paths, such that an ECTL*formula, namely, the negation of the formula $\alpha$ holds in $\mathcal{M}$. In such a case it means that $\mathcal{M}$ violates the property $\alpha$. SAT-based BMC is performed by generating a propositional formula which is satisfiable if and only if such a number $k$ and a set of $k$-paths exist. This formula is usually obtained by combining an encoding of the transition relation of the model and a translation of the negation of $\alpha$. If for a given $k$ this propositional formula is not satisfiable, then $k$ is incremented until either the problem becomes intractable due to the complexity of solving the corresponding SAT instance or $k$ reaches the size of the model. The main advantage of BMC comparing to BDDs is its space efficiency. Moreover, BMC produces counterexamples of minimal length, which eases their interpretation and understanding for debugging purposes.

In order to use the BMC method one needs to define a translation from a given temporal logic to the satisfiability problem (in short: to SAT). There are several translations proposed in literature for different kinds of temporal logics.

The first translation from LTL to SAT was introduced in [2] and another ones in [9] and [3]. The first translation from ECTL to SAT was introduced in [10] and then it was substantially improved in [12]. The first correct translation from ECTL* to SAT was introduced in [11]. A brief description of the idea of this translation is as follows: given an ECTL* formula $\varphi$ and a natural number $k$, first the the number of $k$-paths (paths of length $k$) sufficient for checking the formula $\varphi$ is computed by using a function $f_k$. Next, the set of $f_k(\varphi)$ $k$-paths is created and this set is used to translate every

CONCURRENCY, SPECIFICATION AND PROGRAMMING
M. Szczuka et al. (eds.): Proceedings of the international workshop CS&P 2011
September 28-30, Pułtusk, Poland, pp. 589-600

subformula of the formula $\varphi$. This means that the translation uses all the $f_k(\varphi)$ $k$-paths both for the formula $\varphi$ and for each of its proper subformulae.

In this paper we introduce a new translation from $\mathrm{ECTL}^*$ to SAT. The main idea of our translation consists in translating every subformula $\psi$ of $\varphi$ using only $f_k(\psi)$ of $k$-paths. More precisely, given a formula $\varphi$ and a set $A$ of $k$-paths such that $|A| = f_k(\varphi)$ we divide the set $A$ into subsets needed for translating the subformulae of $\varphi$. In particular, in order to translate a temporal subformula $\psi_1 \mathbf{O} \psi_2$, where $\mathbf{O} \in \{\mathbf{U}, \mathbf{R}\}$, on a given $k$-path $\pi$, we use $\pi$ and exactly $f_k(\psi_1 \mathbf{O} \psi_2)$ $k$-paths for translating the subformula $\psi_1 \mathbf{O} \psi_2$, whereas the old translation uses all the $f_k(\varphi)$ $k$-paths for translating the subformula in question. Generally, if $\psi_1 \mathbf{O} \psi_2$ is a subformulae of $\varphi$, then $f_k(\psi_1 \mathbf{O} \psi_2) < f_k(\varphi)$.

Another novelties provided in this paper are: a simpler definition of bounded semantics and the treatment of the so called loops, which results in significantly simpler propositional formulae being the results of the translation. In particular, unlike the method described in [11], our translation does not require that either all the $k$-paths used in the translation are loops or none is a loop.

We provide some experimental results which clearly confirm that the improvement in question substantially increases the efficiency of the bounded model checking of $\mathrm{ECTL}^*$ formulae.

The rest of the paper is organised as follows. The next section recalls the syntax and (both the bounded and unbounded) semantics of $\mathrm{ECTL}^*$. In Section 3 our improved translation from $\mathrm{ECTL}^*$ to SAT is defined. A sample experimental results are described in Section 4. Section 5 provides concluding remarks.

## 2 Syntax and Semantics of ECTL*

In this section we briefly recall the syntax and semantics of the logic $\mathrm{ECTL}^*$. For more thorough description one can see [1].

The Existential Computation Tree Logic $\mathrm{ECTL}^*$ is a restriction of a propositional branching-time temporal logic $\mathrm{CTL}^*$ that was introduced by Emerson and Halpern in [7] as a specification language for finite-state systems. The restriction consists in using only existential path quantifiers and allowing the negation to be applied to propositional variables only (instead of to arbitrary formulae).

### 2.1 Syntax of ECTL*

The language of $\mathrm{ECTL}^*$ consists of two types of formulae: state formulae (interpreted at states) and path formulae (interpreted along paths). The syntax of $\mathrm{ECTL}^*$ formulae is defined by the following rules, where $AP$ is an arbitrary set of atomic propositions:

- $\mathbf{true}$ and $\mathbf{false}$ are state formulae,
- if $p \in AP$, then $p$ and $\neg p$ are state formulae,
- if $\alpha$ and $\beta$ are state formulae, then $\alpha \wedge \beta$ and $\alpha \vee \beta$ are state formulae,
- if $\varphi$ is a path formula, then $\mathbf{E}\varphi$ is a state formula,
- if $\alpha$ is a state formula, then $\alpha$ is also a path formula,

- if $\varphi$ and $\psi$ are path formulae, then $\varphi \wedge \psi$, $\varphi \vee \psi$, $\mathbf{X}\varphi$, $\varphi\mathbf{U}\psi$ and $\varphi\mathbf{R}\psi$ are path formulae.

In practice, many interesting temporal properties are formulated by using temporal operators $\mathbf{F}$ (eventually) and $\mathbf{G}$ (always) defined as follows: $\mathbf{F}\psi \stackrel{df}{=} \mathbf{true}\,\mathbf{U}\,\psi$, and $\mathbf{G}\psi \stackrel{df}{=} \mathbf{false}\,\mathbf{R}\,\psi$.

### 2.2 Semantics of ECTL*

The semantics of ECTL* formulae is determined with respect to a transition system (see [1]).

**Definition 1.** *A* transition system *is a tuple $\mathcal{M} = (S, Act, \longrightarrow, I, AP, L)$, where $S$ is a nonempty finite set of states, $Act$ is a set of actions, $I \subseteq S$ is a set of initial states, $\longrightarrow \subseteq S \times Act \times S$ is a transition relation, $AP$ is a set of atomic propositions, and $L : S \to 2^{AP}$ is a* labelling function *that assigns to each state a set of atomic propositions that are assumed to be true at that state. Transition systems are also called* models.

For convenience, we shall write $s \stackrel{\sigma}{\longrightarrow} s'$ instead of $(s, \sigma, s') \in \longrightarrow$. Moreover, we shall write $s \longrightarrow s'$ if $s \stackrel{\sigma}{\longrightarrow} s'$ for some $\sigma \in Act$.

In the following it is assumed that a transition system has no terminal states, i.e. for every $s \in S$ there exist $s' \in S$ such that $s \longrightarrow s'$. The set of all natural numbers is denoted by $\mathbb{N}$ and the set of all positive natural numbers by $\mathbb{N}_+$. A *path* in $\mathcal{M}$ is an infinite sequence $\pi = (s_0, s_1, \ldots)$ of states such that $s_j \longrightarrow s_{j+1}$ for each $j \in \mathbb{N}$. For a path $\pi = (s_0, s_1, \ldots)$, let $\pi(j) = s_j$ and $\pi^j = (s_j, s_{j+1}, \ldots)$, for each $j \in \mathbb{N}$. Note that if $\pi$ is a path in $\mathcal{M}$ then the suffix $\pi^j$ is also a path in $\mathcal{M}$. The set of all the paths in a model $\mathcal{M}$ will be denoted by $\Pi(\mathcal{M})$.

Let $\mathcal{M} = (S, Act, \longrightarrow, I, AP, L)$ be a model, $s$ be a state and $\pi$ be a path. For a state formula $\alpha$ over $AP$, the notation $\mathcal{M}, s \models \alpha$ means that $\alpha$ holds at the state $s$ in the model $\mathcal{M}$. Similarly, for a path formula $\varphi$ over $AP$, the notation $\mathcal{M}, \pi \models \varphi$ means that $\varphi$ holds along the path $\pi$ in the model $\mathcal{M}$.

**Definition 2.** *Let $p$ be an atomic proposition, $\alpha$, $\beta$ be state formulae of* ECTL*, *and $\varphi$, $\psi$ be path formulae of* ECTL*. *The relation $\models$ is defined inductively as follows:*

$\mathcal{M}, s \models \mathbf{true}$,
$\mathcal{M}, s \not\models \mathbf{false}$,
$\mathcal{M}, s \models p$ iff $p \in L(s)$,
$\mathcal{M}, s \models \neg p$ iff $p \notin L(s)$,
$\mathcal{M}, s \models \alpha \wedge \beta$ iff $\mathcal{M}, s \models \alpha$ and $\mathcal{M}, s \models \beta$,
$\mathcal{M}, s \models \alpha \vee \beta$ iff $\mathcal{M}, s \models \alpha$ or $\mathcal{M}, s \models \beta$,
$\mathcal{M}, s \models \mathbf{E}\varphi$ iff $\exists \pi \in \Pi(\mathcal{M})\ \big(\pi(0) = s$ and $\mathcal{M}, \pi \models \varphi\big)$,
$\mathcal{M}, \pi \models \alpha$ iff $\mathcal{M}, \pi(0) \models \alpha$,
$\mathcal{M}, \pi \models \varphi \wedge \psi$ iff $\mathcal{M}, \pi \models \varphi$ and $\mathcal{M}, \pi \models \psi$,
$\mathcal{M}, \pi \models \varphi \vee \psi$ iff $\mathcal{M}, \pi \models \varphi$ or $\mathcal{M}, \pi \models \psi$,
$\mathcal{M}, \pi \models \mathbf{X}\varphi$ iff $\mathcal{M}, \pi^1 \models \varphi$,

$$\mathcal{M},\pi \models \varphi\,\mathbf{U}\,\psi \;\text{ iff }\; (\exists j \geqslant 0)\big(\mathcal{M},\pi^j \models \psi \text{ and } (\forall 0 \leqslant i < j)\mathcal{M},\pi^i \models \varphi\big),$$
$$\mathcal{M},\pi \models \varphi\,\mathbf{R}\,\psi \;\text{ iff }\; (\exists j \geqslant 0)\big(\mathcal{M},\pi^j \models \varphi \text{ and } (\forall 0 \leqslant i \leqslant j)\ \mathcal{M},\pi^i \models \psi\big) \text{ or } (\forall j \geqslant 0)\ \mathcal{M},\pi^j \models \psi.$$

*Note 1.* From Definition 2 and the definitions of the operators $\mathbf{F}$ and $\mathbf{G}$ it follows that
$\mathcal{M},\pi \models \mathbf{F}\psi$ iff $(\exists j \geqslant 0)\ \mathcal{M},\pi^j \models \psi$, $\mathcal{M},\pi \models \mathbf{G}\psi$ iff $(\forall j \geqslant 0)\ \mathcal{M},\pi^j \models \psi$.

**Definition 3.** *An* ECTL$^*$ *state formula $\alpha$ is valid in $\mathcal{M}$, denoted by $\mathcal{M} \models \alpha$, iff for each $s \in I$, $\mathcal{M}, s \models \alpha$, i.e., $\alpha$ holds at every initial state of $\mathcal{M}$.*

### 2.3 Bounded Semantics of ECTL*

In this section we define a *bounded semantics* of ECTL$^*$. This is done in order to define the *bounded model checking problem* for ECTL$^*$ and to translate it into the satisfiability problem.

From now on we assume that models are finite, i.e. the sets $S$, $Act$ and $AP$ are finite. To define the bounded semantics one needs to represent infinite paths in a model in a special way. To this aim, we define the notions of $k$*-paths* and *loops*.

**Definition 4.** *Let $\mathcal{M}$ be a model, $k \in \mathbb{N}$, and $0 \leqslant l \leqslant k$. A* $k$-path *is a pair $(\pi, l)$, also denoted by $\pi_l$, where $\pi$ is a finite sequence $\pi = (s_0, \ldots, s_k)$ of states such that $s_j \longrightarrow s_{j+1}$ for each $0 \leqslant j < k$. A $k$-path $\pi_l$ is a* loop *if $l < k$ and $\pi(k) = \pi(l)$.*

The set of all $k$-paths in a model $\mathcal{M}$ will be denoted by $\Pi_k(\mathcal{M})$. Note that the set $\Pi_k(\mathcal{M})$ is finite since $\mathcal{M}$ is finite. Although every $k$-path $\pi_l$ is finite, it still can represent an infinite path if it is a loop. Namely, $\pi_l$ represents the infinite path of the form $uv^\omega$, where $u = (\pi(0), \ldots, \pi(l-1))$ and $v = (\pi(l), \ldots, \pi(k-1))$.

As in the definition of semantics one needs to define the satisfiability relation on suffixes of $k$ paths, we denote by $\pi_l^m$ the pair $(\pi_l, m)$, i.e. the $k$-path $\pi_l$ together with the designated starting point $m$, where $0 \leqslant m \leqslant k$.

Let $s$ be a state and $\pi_l$ be a $k$-path. For a state formula $\alpha$ over $AP$, the notation $\mathcal{M}, s \models_k \alpha$ means that $\alpha$ $k$-holds at the state $s$ in the model $\mathcal{M}$. Similarly, for a path formula $\varphi$ over $AP$, the notation $\mathcal{M}, \pi_l^m \models_k \varphi$, where $0 \leqslant m \leqslant k$, means that $\varphi$ $k$-holds along the suffix $\big(\pi(m), \ldots, \pi(k)\big)$ of $\pi$. For convenience, in the following definition we shall write $s \models_k \alpha$ instead of $\mathcal{M}, s \models_k \alpha$ and $\pi_l^m \models_k \varphi$ instead of $\mathcal{M}, \pi_l^m \models_k \varphi$.

**Definition 5 (Bounded semantics).** *Let $p$ be an atomic proposition, $\alpha$, $\beta$ be state formulae of* ECTL$^*$*, and $\varphi$, $\psi$ be path formulae of* ECTL$^*$*. The relation $\models_k$ is defined inductively as follows:*

$s \models_k \mathbf{true}$,
$s \not\models_k \mathbf{false}$,
$s \models_k p$ iff $p \in L(s)$,
$s \models_k \neg p$ iff $p \notin L(s)$,
$s \models_k \alpha \wedge \beta$ iff $s \models_k \alpha$ and $s \models_k \beta$,
$s \models_k \alpha \vee \beta$ iff $s \models_k \alpha$ or $s \models_k \beta$,

$$
\begin{array}{lll}
s \models_k \mathbf{E}\varphi & \text{iff} & \exists \pi_l \in \Pi_k(\mathcal{M})\ \big(\pi(0) = s \text{ and } \pi_l^0 \models_k \varphi\big), \\
\pi_l^m \models_k \alpha & \text{iff} & \pi(m) \models_k \alpha, \\
\pi_l^m \models_k \varphi \wedge \psi & \text{iff} & \pi_l^m \models_k \varphi \text{ and } \pi_l^m \models_k \psi, \\
\pi_l^m \models_k \varphi \vee \psi & \text{iff} & \pi_l^m \models_k \varphi \text{ or } \pi_l^m \models_k \psi, \\
\pi_l^m \models_k \mathbf{X}\varphi & \text{iff} & \big(m < k \text{ and } \pi_l^{m+1} \models_k \alpha\big) \\
 & & \text{or } \big(m = k \text{ and } \pi(k) = \pi(l) \text{ and } \pi_l^{l+1} \models_k \alpha\big), \\
\pi_l^m \models_k \varphi\,\mathbf{U}\,\psi & \text{iff} & (\exists m \leqslant j \leqslant k)\big(\pi_l^j \models_k \psi \text{ and } (\forall m \leqslant i < j)\pi_l^i \models_k \varphi\big) \\
 & & \text{or } \big(l < m \text{ and } \pi(k) = \pi(l) \text{ and } (\exists l < j < m)\ \pi_l^j \models_k \psi \\
 & & \quad \text{and } (\forall l < i < j)\ \pi_l^i \models \varphi \text{ and } (\forall m \leqslant i \leqslant k)\ \pi_l^i \models_k \varphi\big), \\
\pi_l^m \models_k \varphi\,\mathbf{R}\,\psi & \text{iff} & (\exists m \leqslant j \leqslant k)\big(\pi_l^j \models_k \varphi \text{ and } (\forall m \leqslant i \leqslant j)\pi_l^i \models_k \psi\big) \\
 & & \text{or } \big(l < m \text{ and } \pi(k) = \pi(l) \text{ and } (\exists l < j < m)\ \pi_l^j \models_k \varphi \\
 & & \quad \text{and } (\forall l < i \leqslant j)\ \pi_l^i \models \psi \text{ and } (\forall m \leqslant i \leqslant k)\ \pi_l^i \models_k \psi\big) \\
 & & \text{or } \big(l < k \text{ and } \pi(k) = \pi(l) \text{ and } (\forall min(m,l) \leqslant j \leqslant k)\ \pi_l^j \models_k \psi\big).
\end{array}
$$

*Note 2.* From Definition 5 and the definitions of the operators $\mathbf{F}$ and $\mathbf{G}$ it follows that

$$
\begin{array}{lll}
\pi_l^m \models_k \mathbf{F}\psi & \text{iff} & (\exists m \leqslant j \leqslant k)\ \pi_l^j \models_k \psi \\
 & & \text{or } \big(l < m \text{ and } \pi(k) = \pi(l) \text{ and } (\exists l < j < m)\ \pi_l^j \models_k \psi\big), \\
\pi_l^m \models_k \mathbf{G}\psi & \text{iff} & l < k \text{ and } \pi(k) = \pi(l) \text{ and } (\forall min(m,l) \leqslant j \leqslant k)\ \pi_l^j \models_k \psi.
\end{array}
$$

*Example 1.* Let $\mathcal{M}_1 = (\{a,b,c\}, \{\tau\}, \longrightarrow, \{a\}, \{p\}, L)$ be a transition system such that $\longrightarrow = \{(a,\tau,b), (b,\tau,a), (a,\tau,c), (c,\tau,a)\}$, $L(a) = L(c) = \{p\}$ and $L(b) = \emptyset$. Consider the path formula $\varphi = \mathbf{XXG}p$ and the sequence $\pi = (a,b,a,c,a)$. It is easy to check that $\mathcal{M}_1, \pi_0^0 \not\models_4 \varphi$ but $\mathcal{M}_1, \pi_2^0 \models_4 \varphi$.

*Example 2.* Let $\mathcal{M}_2 = (\{a,b\}, \{\tau\}, \longrightarrow, \{a\}, \{p,q\}, L)$ be a transition system such that $\longrightarrow = \{(a,\tau,b), (b,\tau,a)\}$, $L(a) = \{p\}$ and $L(b) = \{q\}$. Notice that the only initial path in $\mathcal{M}_2$ is of the form $(ab)^\omega$.

It is easily seen that the state formula $\alpha = \mathbf{EGX}(p \vee q)$ holds in $\mathcal{M}_2$. In order to check that $\alpha$ 2-holds in $\mathcal{M}_2$ observe that $\mathcal{M}_2, a \models_2 \alpha$ as $\mathcal{M}_2, \pi_0^0 \models_2 \mathbf{GX}(p \vee q)$, where $\pi = (a,b,a)$. This example shows that the second component of the condition for the bounded satisfiability of $\mathbf{X}\varphi$ cannot be avoided.

It is easily seen that the state formula $\beta = \mathbf{E}(\mathbf{GF}p \wedge \mathbf{GF}q)$ holds in $\mathcal{M}_2$. In order to check that $\beta$ 2-holds in $\mathcal{M}_2$ observe that $\mathcal{M}_2, a \models_2 \beta$ as $\mathcal{M}_2, \pi_0^0 \models_2 \mathbf{GF}p \wedge \mathbf{GF}q$, where $\pi = (a,b,a)$. This example shows that the second component of the condition for the bounded satisfiability of $\varphi\mathbf{U}\psi$ also cannot be avoided.

**Definition 6.** *An* ECTL$^*$ *state formula $\alpha$ is $k$-valid in $\mathcal{M}$, denoted by $\mathcal{M} \models_k \alpha$, iff for each $s \in I$, $\mathcal{M}, s \models_k \alpha$, i.e., $\alpha$ $k$-holds at every initial state of $\mathcal{M}$.*

The following theorem states that for a given model $\mathcal{M}$ and a formula $\alpha$ there exists a bound $k$ such that the model checking problem $\mathcal{M} \models \alpha$ can be reduced to the bounded model checking problem $\mathcal{M} \models_k \alpha$.

**Theorem 1.** *Let $\mathcal{M}$ be a model and $\alpha$ be an* ECTL$^*$ *state formula. Then $\mathcal{M} \models \alpha$ iff for some $k \in \mathbb{N}$, $\mathcal{M} \models_k \alpha$.*

The proof of Theorem 1 will be provided in the full version of the present paper.

## 3 Translation to SAT

First, we briefly describe how to translate the problem of $k$-validity of an $\mathrm{ECTL}^*$ formula to the problem of satisfiability of some propositional formula. Next, we describe the new translation from $\mathrm{ECTL}^*$ formulae to propositional formulae built over a set $PV$ of propositional variables plus the constants $\mathbf{true}$ and $\mathbf{false}$, with help of the propositional connectives $\neg$, $\wedge$, $\vee$ and $\rightarrow$.

### 3.1 Boolean encoding

We begin with the encoding of the transitions of the given model $\mathcal{M}$. Since the set of states of $\mathcal{M}$ is finite, every element of $S$ can be encoded as a bit vector of some length $r$ depending on the number of states of $\mathcal{M}$. Then, each state of $\mathcal{M}$ can be represented by a valuation of a vector $\mathbf{w} = (w_{j_1}, \ldots, w_{j_r})$ (called a *symbolic state*) of different propositional variables called *propositional state variables*.

The designated position $l$ of any $k$-path can be encoded as a bit vector of the length $t = \max(1, \lceil \log_2(k+1) \rceil)$. Therefore, such a position $l$ of the path of the number $j$ can be represented by a valuation of a vector $\mathbf{u} = (u_{j_1}, \ldots, u_{j_t})$ (called a *symbolic number*) of different propositional variables, called *propositional natural variables*, not appearing among propositional state variables. In the following we denote by $Var(\mathbf{w})$ the set of propositional state variables appearing in $\mathbf{w}$ and by $Var(\mathbf{u})$ the set of propositional natural variables appearing in $\mathbf{u}$.

From now on let $SV$ be a set of propositional variables used as propositional state variables and $NV$ be a set of propositional variables used as propositional natural variables. We assume that $SV \cap NV = \emptyset$. Let $PV = SV \cup NV$ and $V : PV \rightarrow \{0,1\}$ be a *valuation of propositional variables* (a *valuation* for short). Each valuation $V$ induces the functions $\mathbf{S} : SV^r \rightarrow \{0,1\}^r$ and $\mathbf{J} : NV^t \rightarrow \mathbb{N}$ defined in the following way:

$$\mathbf{S}(w_{j_1}, \ldots, w_{j_r}) = (V(w_{j_1}), \ldots, V(w_{j_r})), \quad \mathbf{J}((u_{j_1}, \ldots, u_{j_t})) = \sum_{i=1}^{t} V(u_{j_i}) \cdot 2^{i-1}.$$

A pair consisting of a finite sequence of length $k$ of symbolic states and of a symbolic number is called a *symbolic $k$-path*. In order to translate an $\mathrm{ECTL}^*$ formula $\varphi$ one needs to use not just one but exactly $f_k(\varphi)$ symbolic $k$-paths, where $f_k$ is a function defined below. In the following, $F_k(\varphi)$ denotes the set $\{j \in \mathbb{N} \mid 1 \leqslant j \leqslant f_k(\varphi)\}$.

Now, for a given $k \in \mathbb{N}$, let $\mathbf{W}_{r,k}$ be a fixed finite set of global state variables $\mathbf{w}_{m,n} \in PV^r$, where $(m,n) \in \{0,0\} \cup \{0,\ldots,k\} \times F_k(\varphi)$, such that $Var(\mathbf{w}_{m,n}) \cap Var(\mathbf{w}_{m',n'}) = \emptyset$, provided $(m,n) \neq (m',n')$. Moreover, let $\mathbf{N}_k$ be a fixed finite set of natural variables $\mathbf{u}_j \in PV^t$, where $j \in F_k(\varphi)$, such that $Var(\mathbf{u}_j) \cap Var(\mathbf{u}_{j'}) = \emptyset$, provided $j \neq j'$. We define for each $j \in F_k(\varphi)$ the $j$-th symbolic $k$-path as $\boldsymbol{\pi}_j = \big((\mathbf{w}_{0,j}, ..., \mathbf{w}_{k,j}), \mathbf{u}_j\big)$.

Furthermore, let $\mathbf{w}$ and $\mathbf{w}'$ be symbolic states from $\mathbf{W}_{r,k}$, $\mathbf{u}$ be a symbolic number from $\mathbf{N}_k$, and $\boldsymbol{\pi}_j$ be the $j$-th symbolic $k$-path. Given a model $\mathcal{M}$ we define the following propositional formulae:

- $\mathcal{P}_p(\mathbf{w})$, where $p \in AP$, as a formula such that for each $V \in \{0,1\}^{PV}$, $V$ satisfies $P_p(\mathbf{w})$ iff $p \in L(\mathbf{S}(\mathbf{w}))$,
- $\mathcal{I}(\mathbf{w})$ as a formula such that for each $V \in \{0,1\}^{PV}$, $V$ satisfies $\mathcal{I}(\mathbf{w})$ iff $\mathbf{S}(\mathbf{w}) \in I$,

- $\mathcal{T}(\mathbf{w}, \mathbf{w}')$ as a formula such that for each $V \in \{0,1\}^{PV}$, $V$ satisfies $\mathcal{T}(\mathbf{w}, \mathbf{w}')$ iff $\mathbf{S}(\mathbf{w}) \longrightarrow \mathbf{S}(\mathbf{w}')$ in $\mathcal{M}$,
- $\mathcal{H}(\mathbf{w}, \mathbf{w}')$ as a formula such that for each $V \in \{0,1\}^{PV}$, $V$ satisfies $\mathcal{H}(\mathbf{w}, \mathbf{w}')$ iff both $\mathbf{S}(\mathbf{w})$ and $\mathbf{S}(\mathbf{w}')$ are states of $\mathcal{M}$ and $\mathbf{S}(\mathbf{w}) = \mathbf{S}(\mathbf{w}')$,
- $\mathcal{B}_j^{\sim}(\mathbf{u})$ as a formula such that for each $V \in \{0,1\}^{PV}$, $V$ satisfies $\mathcal{B}_j^{\sim}(\mathbf{u})$ iff $j \sim \mathbf{J}(\mathbf{u})$, where $\sim \in \{<, \leqslant, =, \geqslant, >\}$,
- $\mathcal{L}_k^l(\boldsymbol{\pi}_n)$ as a formula $\mathcal{H}(\mathbf{w}_{k,n}, \mathbf{w}_{l,n}) \wedge \mathcal{B}_l^{=}(\mathbf{u}_n)$ which is satisfied by a valuation $V$ iff $\mathbf{S}(\mathbf{w}_{k,n}) = \mathbf{S}(\mathbf{w}_{l,n})$ and $l = \mathbf{J}(\mathbf{u}_n)$.

### 3.2 The new translation

In order to translate a formula $\varphi$, one needs to know the number of $k$-paths sufficient to translate this formula. To this aim, the function $f_k$, first introduced in [11], is used.

**Definition 7.** *Let $\mathcal{F}_p$ be the set of all the* ECTL* *path formulae built over a given set $AP$ of atomic propositions. The function $f_k : \mathcal{F}_p \to \mathbb{N}$ is defined as follows:*

$$f_k(\mathbf{true}) = 0 \qquad f_k(\mathbf{false}) = 0$$
$$f_k(\varphi) = 0, \textit{if } \varphi \in AP \qquad f_k(\neg\varphi) = 0, \textit{if } \varphi \in AP$$
$$f_k(\varphi \vee \psi) = \max\{f_k(\varphi), f_k(\psi)\} \qquad f_k(\varphi \wedge \psi) = f_k(\varphi) + f_k(\psi)$$
$$f_k(\mathbf{E}\varphi) = f_k(\varphi) + 1 \qquad f_k(\mathbf{X}\varphi) = f_k(\varphi)$$
$$f_k(\varphi \mathbf{U} \psi) = k \cdot f_k(\varphi) + f_k(\psi) \qquad f_k(\varphi \mathbf{R} \psi) = (k+1) \cdot f_k(\psi) + f_k(\varphi)$$

For every ECTL* path formula $\varphi$ the function $f_k$ determines how many symbolic $k$-paths are needed for translating the formula $\varphi$. Notice that the translation from [11] translates every subformula of the form $\mathbf{E}\psi$ using all the $f_k(\psi)$ $k$-paths.

Our improvement consists in translating every subformula $\psi$ of $\varphi$ by using only $f_k(\psi)$ $k$-paths. For this, given a formula $\varphi$ and a set $A$ of $k$-paths such that $|A| = f_k(\varphi)$, we divide the set $A$ into subsets needed for translating the subformulae of $\varphi$. In particular, for translating a temporal subformula of the form $\mathbf{E}\psi$ we choose exactly one path for translating the operator $\mathbf{E}$ and use the remaining $k$-paths for translating $\psi$.

In order to divide $A$ into proper subsets we need some auxiliary functions. In the definitions of these functions we will use the relation $\prec$ defined on the set of all finite subsets of $\mathbb{N}$ in the following way: $A \prec B$ iff for all natural numbers $x$ and $y$, $x \in A$ and $y \in B$ implies $x < y$. Note that from the definition of $\prec$ it follows that $A \prec B$ iff either $A = \emptyset$ or $B = \emptyset$ or $A \neq \emptyset$, $B \neq \emptyset$, $A \cap B = \emptyset$ and $\max(A) < \min(B)$.

In order to explain the purpose of the auxiliary functions defined above let us recall that each ECTL* path formula $\varphi$ will be translated by using a set of exactly $f_k(\varphi)$ symbolic $k$-paths. In the following we will say that a set $A$ of positive natural numbers is used to translate a formula $\varphi$ instead of saying that the set of symbolic $k$-paths whose numbers are in $A$ is used to translate the formula $\varphi$. Moreover, saying that a set $A$ is used to translate a formula $\varphi$ assumes that $|A| = f_k(\varphi)$.

Now, let $A$ be a finite nonempty set of natural numbers, and $k$, $p$ be natural numbers with $p \leqslant |A|$. Then,

- $g_l(A, p)$ denotes the subset $B$ of $A$ such that $|B| = p$ and $B \prec A \setminus B$,
- $g_r(A, p)$ denotes the subset $C$ of $A$ such that $|C| = p$ and $A \setminus C \prec C$,
- $g_s(A)$ denotes the set $A \setminus \{\min(A)\}$,

- if $k$ divides $|A| - p$, then $h^U_k(A,p)$ denotes the sequence $(B_0,\ldots,B_k)$ of subsets of $A$ such that $\bigcup_{j=0}^{k} B_j = A$, $|B_0| = \ldots = |B_{k-1}|$, $|B_k| = p$, and $B_i \prec B_j$ for every $0 \leqslant i < j \leqslant k$,
- if $k+1$ divides $|A| - p$, then $h^R_k(A,p)$ denotes the sequence $(B_0,\ldots,B_{k+1})$ of subsets of $A$ such that $\bigcup_{j=0}^{k+1} B_j = A$, $|B_0| = \ldots = |B_k|$, $|B_{k+1}| = p$, and $B_i \prec B_j$ for every $0 \leqslant i < j \leqslant k$.

If $h^U_k(A,p) = (B_0,\ldots,B_k)$, then $h^U_k(A,p)(j)$ denotes the set $B_j$, for every $0 \leqslant j \leqslant k$. Similarly, if $h^R_k(A,p) = (B_0,\ldots,B_{k+1})$, then $h^R_k(A,p)(j)$ denotes the set $B_j$, for every $0 \leqslant j \leqslant k+1$.

The functions $g_l$ and $g_r$ are used in the translation of the formulae with the main connective being either conjunction or disjunction: for a given ECTL$^*$ formula $\varphi \wedge \psi$, if a set $A$ is to be used to translate this formula, then the set $g_l(A, f_k(\varphi))$ is used to translate the subformula $\varphi$ and the set $g_r(A, f_k(\psi))$ is used to translate the subformula $\psi$; for a given ECTL$^*$ formula $\varphi \vee \psi$, if a set $A$ is to be used to translate this formula, then the set $g_l(A, f_k(\varphi))$ is used to translate the subformula $\varphi$ and the set $g_l(A, f_k(\psi))$ is used to translate the subformula $\psi$.

The function $g_s$ is used in the translation of the subformulae with the main connective $\mathbf{E}$: for a given ECTL$^*$ formula of the form $\mathbf{E}\,\varphi$, if a set $A$ is to be used to translate this formula, then the symbolic path $\boldsymbol{\pi}_{\min(A)}$ is used to translate the operator $\mathbf{E}$ and the set $g_s(A)$ is used to translate the subformula $\varphi$ at the symbolic path $\boldsymbol{\pi}_{\min(A)}$.

The function $h^U_k$ is used in the translation of subformulae of the form $\varphi\,\mathbf{U}\,\psi$: if a set $A$ is to be used to translate the subformula $\varphi\,\mathbf{U}\,\psi$ at the symbolic $k$-path $\boldsymbol{\pi}_n$ (with starting point $m$), then for every $i$ such that $m \leqslant j < i$, the set $h^U_k(A, f_k(\psi))(k)$ is used to translate the formula $\psi$ at the symbolic path $\boldsymbol{\pi}_n$ with starting point $i$; moreover, for every $j$ such that $m \leqslant j < i$, the set $h^U_k(A, f_k(\psi))(j)$ is used to translate the formula $\varphi$ at the symbolic path $\boldsymbol{\pi}_n$ with starting point $j$. Notice that if $k$ does not divide $|A|-p$, then $h^U_k(A,p)$ is undefined. However, for every set $A$ such that $|A| = f_k(\varphi\,\mathbf{U}\,\psi)$, it is clear from the definition of $f_k$ that $k$ divides $|A| - f_k(\psi)$.

The function $h^R_k$ is used in the translation of subformulae of the form $\varphi\,\mathbf{R}\,\psi$: if a set $A$ is to be used to translate the subformula $\varphi\,\mathbf{R}\,\psi$ at a symbolic $k$-path $\boldsymbol{\pi}_n$ (with starting point $m$), then for every $i$ such that $m \leqslant i \leqslant k$, the set $h^R_k(A, f_k(\varphi))(k)$, is used to translate the formula $\varphi$ at the symbolic paths $\boldsymbol{\pi}_n$ with starting point $i$; moreover, for every $j$ such that $m \leqslant j \leqslant i$, the set $h^R_k(A, f_k(\varphi))(j)$ is used to translate the formula $\psi$ at the symbolic path $\boldsymbol{\pi}_n$ with starting point $j$. Notice that if $k+1$ does not divide $|A|-1$, then $h^R_k(A,p)$ is undefined. However, for every set $A$ such that $|A| = f_k(\varphi\,\mathbf{R}\,\psi)$, it is clear from the definition of $f_k$ that $k+1$ divides $|A| - f_k(\varphi)$.

Now we are ready to define our translation. Let $\alpha$ be an ECTL$^*$ state formula and $A \subset \mathbb{N}_+$ be a set of numbers of symbolic $k$-paths such that $|A| = f_k(\alpha)$, $\varphi$ be an ECTL$^*$ formula and $B \subset \mathbb{N}_+$ be a set of numbers of symbolic $k$-paths such that $|B| = f_k(\varphi)$, $n \in \mathbb{N}$, and $0 \leqslant m \leqslant k$. By $\langle\alpha\rangle^{[m,n,A]}_k$ we denote the translation of an ECTL$^*$ state formula $\alpha$ at the symbolic state $\mathbf{w}_{m,n}$ by using the set $A$, and by $[\varphi]^{[m,n,B]}_k$ we denote the translation of an ECTL$^*$ path formula $\varphi$ on the symbolic $k$-path $\boldsymbol{\pi}_n$ with starting point $m$ by using the set $B$.

**Translation of ECTL* formulae**

$$
\begin{aligned}
\langle \mathbf{true} \rangle_k^{[m,n,A]} &:= \mathbf{true} \\
\langle \mathbf{false} \rangle_k^{[m,n,A]} &:= \mathbf{false} \\
\langle p \rangle_k^{[m,n,A]} &:= P_p(\mathbf{w}_{m,n}) \\
\langle \neg p \rangle_k^{[m,n,A]} &:= \neg P_p(\mathbf{w}_{m,n}) \\
\langle \alpha \diamond \beta \rangle_k^{[m,n,A]} &:= \langle \alpha \rangle_k^{[m,n,g_l(A,f_k(\alpha))]} \diamond \langle \beta \rangle_k^{[m,n,g_r(A,f_k(\beta))]}, \text{where } \diamond \in \{\wedge, \vee\} \\
\langle \mathbf{E}\varphi \rangle_k^{[m,n,A]} &:= H(\mathbf{w}_{m,n}, \mathbf{w}_{0,\min(A)}) \wedge \bigvee_{l=0}^{k} B_l^{=}(\mathbf{u}_{\min(A)}) \wedge [\varphi]_k^{[0,\min(A),g_s(A)]} \\
[\alpha]_k^{[m,n,A]} &:= \langle \alpha \rangle_k^{[m,n,A]} \\
[\varphi \diamond \psi]_k^{[m,n,A]} &:= [\varphi]_k^{[m,n,g_l(A,f_k(\varphi))]} \diamond [\psi]_k^{[m,n,g_r(A,f_k(\psi))]}, \text{where } \diamond \in \{\wedge, \vee\} \\
[\mathbf{X}\varphi]_k^{[m,n,A]} &:= \begin{cases} [\varphi]_k^{[m+1,n,A]}, & \text{if } m < k \\ \bigvee_{l=0}^{k-1} \left( \mathcal{L}_k^l(\boldsymbol{\pi}_n) \wedge [\varphi]_k^{[l+1,n,A]} \right), & \text{if } m = k \end{cases} \\
[\varphi \,\mathbf{U}\, \psi)]_k^{[m,n,A]} &:= \bigvee_{j=m}^{k} \left( [\psi]_k^{[j,n,h_k^U(A,\, f_k(\psi))(k)]} \wedge \bigwedge_{i=m}^{j-1} [\varphi]_k^{[i,n,h_k^U(A,f_k(\psi))(i)]} \right) \\
&\quad \vee \left( \bigvee_{l=0}^{m-1} \left( \mathcal{L}_k^l(\boldsymbol{\pi}_n) \right) \wedge \bigvee_{j=0}^{m-1} \left( \mathcal{B}_j^{>}(\mathbf{u}_n) \wedge [\psi]_k^{[j,n,h_k^U(A,\, f_k(\psi))(k)]} \right.\right. \\
&\qquad \left. \wedge \bigwedge_{i=0}^{j-1} \left( \mathcal{B}_i^{>}(\mathbf{u}_n) \rightarrow [\varphi]_k^{[i,n,h_k^U(A,f_k(\psi))(i)]} \right) \right. \\
&\qquad \left. \left. \wedge \bigwedge_{i=m}^{k} [\varphi]_k^{[i,n,h_k^U(A,f_k(\psi))(i)]} \right) \right) \\
[\varphi \,\mathbf{R}\, \psi)]_k^{[m,n,A]} &:= \bigvee_{j=m}^{k} \left( [\varphi]_k^{[j,n,h_k^R(A,\, f_k(\varphi))(k)]} \wedge \bigwedge_{i=m}^{i} [\psi]_k^{[i,n,h_k^R(A,f_k(\varphi))(i)]} \right) \\
&\quad \vee \left( \bigvee_{l=0}^{m-1} \left( \mathcal{L}_k^l(\boldsymbol{\pi}_n) \right) \wedge \bigvee_{j=0}^{m} \left( \mathcal{B}_j^{>}(\mathbf{u}_n) \wedge [\varphi_k^{[j,n,h_k^U(A,\, f_k(\psi))(k)]} \right.\right. \\
&\qquad \left. \wedge \bigwedge_{i=0}^{j-1} \left( \mathcal{B}_i^{>}(\mathbf{u}_n) \rightarrow [\psi]_k^{[i,n,h_k^U(A,f_k(\psi))(i)]} \right) \right. \\
&\qquad \left. \left. \wedge \bigwedge_{i=m}^{k} [\psi]_k^{[i,n,h_k^U(A,f_k(\psi))(i)]} \right) \right) \\
&\quad \vee \left( \bigvee_{l=0}^{k-1} \left( \mathcal{L}_k^l(\boldsymbol{\pi}_n) \right) \wedge \bigwedge_{j=0}^{m-1} \left( \mathcal{B}_j^{\geqslant}(\mathbf{u}_n) \rightarrow [\psi]_k^{[j,n,h_k^R(A,f_k(\varphi))(j)]} \right) \right. \\
&\qquad \left. \wedge \bigwedge_{j=m}^{k} [\psi]_k^{[j,n,h_k^R(A,f_k(\varphi))(j)]} \right)
\end{aligned}
$$

*Note 3.* From the definitions of the operators $\mathbf{F}$ and $\mathbf{G}$ it follows that

$$
\begin{aligned}
[\mathbf{F}\,\psi]_k^{[m,n,\,A]} \;:=\; & \bigvee_{i=m}^{k} [\psi]_k^{[i,n,h_k^U(A,\,f_k(\psi))(k)]} \\
& \vee \left( \mathcal{L}_k(\boldsymbol{\pi}_n) \wedge \bigvee_{i=0}^{m-1} \left( \mathcal{B}_i^{>}(\mathbf{u}_n) \wedge [\psi]_k^{[i,n,h_k^U(A,\,f_k(\psi))(k)]} \right) \right) \\
[\mathbf{G}\,\psi]_k^{[m,n,\,A]} \;:=\; & \bigvee_{l=0}^{k-1} \left( \mathcal{L}_k^l(\boldsymbol{\pi}_n) \right) \wedge \bigwedge_{j=0}^{m-1} \left( \mathcal{B}_j^{\geqslant}(\mathbf{u}_n) \rightarrow [\psi]_k^{[j,n,h_k^R(A,f_k(\varphi))(j)]} \right) \\
& \wedge \bigwedge_{j=m}^{k} [\psi]_k^{[j,n,h_k^R(A,f_k(\varphi))(j)]}.
\end{aligned}
$$

### 3.3 Bounded Model Checking of ECTL* properties

First, define the *unfolding of the transition relation* $[M]_k^A$ for a finite set $A$ of symbolic $k$-paths, where $A = \{j \in \mathbb{N} \mid 1 \leqslant j \leqslant n\}$, for some $n \in \mathbb{N}$.

$$
[M]_k^A := \bigwedge_{j \in A} \bigwedge_{i=0}^{k-1} \mathcal{T}(\mathbf{w}_{i,j}, \mathbf{w}_{i+1,j})
$$

Now let us recall how to verify by the BMC method a given $\mathrm{ECTL}^*$ state formula $\alpha$. For this one has to check the satisfiability of the following conjunction:

$$
[M]_k^\alpha := \mathcal{I}(\mathbf{w}_{0,0}) \wedge [M]_k^{F_k(\alpha)} \wedge \langle \alpha \rangle_k^{[0,0,F_k(\alpha)]}
$$

starting with $k = 0$. If for a given $k$ the formula $[M]_k^\alpha$ is not satisfiable, then $k$ is increased and the resulting formula is to be checked by a SAT-solver again. The method described relies on the following Theorem 2.

**Theorem 2.** *Let $\mathcal{M}$ be a model and $\alpha$ be an* $\mathrm{ECTL}^*$ *state formula. Then for every $k \in \mathbb{N}$, $\mathcal{M} \models_k \alpha$ if, and only if, the propositional formula $[M]_k^\alpha$ is satisfiable.*

The proof of Theorem 2 will be provided in the full version of the present paper.

## 4 Experimental results

In order to compare the new translation with the old one we have implemented both the old translation (as described in [11]) and the new one as the standalone programs written in the programming language C++. Our experiments were performed on a computer equipped with Intel Xeon 2 GHz processor, 4GB of RAM and the operating system Ubuntu Linux Server with the kernel 2.6.38. The both tools and the benchmarks can be found at the webpage `http://ajd.czest.pl/~modelchecking/software.html`, together with an instruction how to repeat the experiments.

As the benchmark we used the well-known dining philosophers problem ([5], [8]). We have modelled this problem by means of communicating finite automata. The system consists of $n$ automata each of which models a philosopher, together with $n$ automata each of which models a fork, together with one automaton which models the lackey. The latter automaton is used to coordinate the philosophers' access to the dining-room. In fact, this automaton ensures that no deadlock is possible. The global system is obtained as the parallel composition of the components, which are shown in Figure 1.

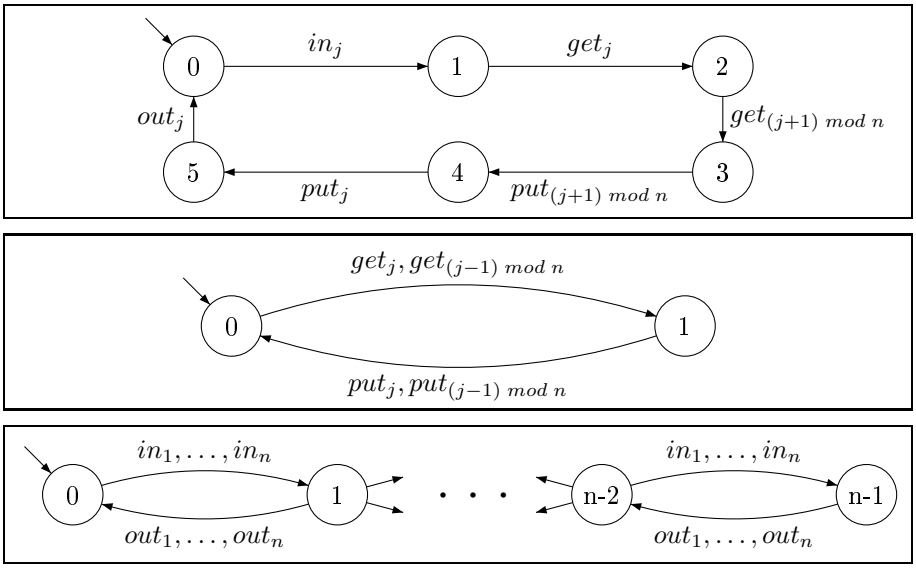


**Fig. 1.** The automata for the j-th Philosopher, the j-th Fork and the Lackey

Let $AP = \{p_j^i \mid 0 \leqslant j < n \ \wedge \ 0 \leqslant i \leqslant 5\}$ and assume that the variable $p_j^i$ is true only at the state $s_j$ of the automaton for the philosopher of number $i$. We have tested the following ECTL* state formula over $AP$:

$$\varphi_n = \bigwedge_{j=0}^{n-1} \mathbf{EF}\big(p_j^2 \wedge \mathbf{E}\big(\mathbf{F}p_j^3 \wedge \mathbf{G}\neg p_j^4\big)\big).$$

in order to compare experimental results both for the old and new translations. This formula expresses the following property:

*For each philosopher there exists a path such that eventually this philosopher has got his left fork and from such a situation there exists a path on which eventually the philosopher has got his right fork and then he is eating forever.*

It turned out that the formula $\varphi_n$ is valid in the considered model for every $n \geq 4$. Moreover, notice that $LL = f_k(\varphi_n) = 2 \cdot n$.

Table 1 shows the experimental results of verification of the formula $\varphi_n$ for different numbers of philosophers using the old and the new translation, respectively. The data in the table are organised as follows: the first column indicates which of the two translations applies to the given row, the second column shows the number of philosophers, the third the length of the witness, and the fourth the value of $f_k(\varphi_n)$. Then, the fifth and sixth columns contains the time and memory consumed by BMC to generate

the set of clauses, while the next two columns give the time and memory consumed by the state of the art SAT solver MiniSAT [6]. The last two columns are self-explanatory.

| | | | | BMC | | MiniSAT | | Total | |
|---|---|---|---|---|---|---|---|---|---|
| **translation** | **n** | **k** | **LL** | **sec** | **MB** | **sec** | **MB** | **sec** | **MB** |
| old | 15 | 7 | 30 | 1073.4 | 206.1 | 22244.9 | 4596.0 | 23318.3 | 4596.0 |
| new | 15 | 7 | 30 | 11.3 | 43.1 | 54.0 | 224.0 | 65.3 | 224.0 |
| new | 35 | 7 | 70 | 126.7 | 340.5 | 3037.5 | 3256.0 | 3164.2 | 3256.0 |
| new | 40 | 7 | 80 | 186.6 | 491.5 | 8329.7 | 5411.0 | 8516.3 | 5411.0 |

Table 1: The results of the old and the new translation of $\varphi_n$.

## 5 Conclusions and Future Work

We have shown that our new translation from $\mathrm{ECTL}^*$ to $\mathrm{SAT}$ results in reducing the effort of the $\mathrm{SAT}$ solver while testing satisfiability of the formula to be verified. The experiments confirm that the improvement in question leads to a reduction of the size of the CNF formulae submitted to the $\mathrm{SAT}$ solver, and therefore to a significant reduction both in the time and memory required by the $\mathrm{SAT}$ solver to return an answer. Additional experiments are necessary to compare the effectiveness of our new translation with the standalone translation from $\mathrm{ECTL}$ to $\mathrm{SAT}$ described in [12] and with various translations for $\mathrm{LTL}$.

## References


1. C. Baier and J.-P. Katoen. *Principles of model checking*. MIT Press, 2008.
2. A. Biere, A. Cimatti, E. Clarke, M.Fujita, and Y. Zhu. Symbolic model checking using SAT procedures instead of BDDs. In *Proc. of the ACM/IEEE Design Automation Conference (DAC'99)*, pages 317–320, 1999.
3. A. Biere, K. Heljanko, T. A. Junttila, T. Latvala, and V. Schuppan. Linear encodings of bounded LTL model checking. *Logical Methods in Computer Science*, 2(5), 2006.
4. E. M. Clarke, D. Kroening, J. Ouaknine, and O. Strichman. Completeness and complexity of bounded model checking. In B. Steffen and G. Levi, editors, *VMCAI*, volume 2937 of *Lecture Notes in Computer Science*, pages 85–96. Springer, 2004.
5. E.W. Dijkstra. Hierarchical ordering of sequential processes. *Acta Inf.*, 1:115–138, 1971.
6. N. Eén and N. Sörensson. MiniSat Page. http://minisat.se/MiniSat.html.
7. E. Allen Emerson and Joseph Y. Halpern. Decision procedures and expressiveness in the temporal logic of branching time. *J. Comput. Syst. Sci.*, 30(1):1–24, 1985.
8. C.A.R. Hoare. *Communicating sequential processes*. Prentice Hall, 1985.
9. T. Latvala, A. Biere, K. Heljanko, and T. A. Junttila. Simple bounded LTL model checking. In A. J. Hu and A. K. Martin, editors, *FMCAD*, volume 3312 of *Lecture Notes in Computer Science*, pages 186–200. Springer, 2004.
10. W. Penczek, B. Woźna, and A. Zbrzezny. Bounded model checking for the universal fragment of CTL. *Fundamenta Informaticae*, 51(1-2):135–156, 2002.
11. B. Woźna. $\mathrm{ACTL}^*$ properties and bounded model checking. *Fundamenta Informaticae*, 63(1):65–87, 2004.
12. A. Zbrzezny. Improving the translation from ECTL to SAT. *Fundamenta Informaticae*, 85(1-4):513–531, 2008.