Model Checking Games for Branching Time Logics

Martin Lange and Colin Stirling

LFCS, Division of Informatics
The University of Edinburgh

email: {martin,cps}@dcs.ed.ac.uk

December 2000

Abstract

This paper defines and examines model checking games for the branching time temporal logic **CTL***. The games employ a technique called focus which enriches sets by picking out one distinguished element. This is necessary to avoid ambiguities in the regeneration of temporal operators. The correctness of these games is proved, and optimisations are considered to obtain model checking games for important fragments of **CTL***. A game based model checking algorithm that matches the known lower and upper complexity bounds is sketched.

1 Introduction

Model checking is a useful and broadly accepted technique for verifying parallel processes. The system to be examined is abstracted into a mathematical interpretation for a logical formula which formalises a property the system is expected to have or to lack. A model checking algorithm decides whether the system's abstraction fulfils the formula and thus whether the system meets its specification given by the formula, provided that the abstraction is correct. We will not discuss the finding of good abstractions at all, instead we are interested in checking properties of the abstraction only. Hence, in the following, the term "system" will denote the abstraction as well.

However, verification of concurrent systems is often combined with specification in the framework of developing them. For such a process a simple yes/no answer to the question whether a system is correct w.r.t. a certain property is not sufficient. Moreover, techniques that show why or where the property is violated are required.

Model checking games being played by two players on the system and the formula provide such features. Answering the question about the property being fulfilled turns out to be equivalent to finding a winning strategy for one of the players. Once such a strategy is found, i.e. computed by a verification tool for example, it can be used to enable an interactive play between the tool and the developer.

There are various classes of interpretations which are suitable for modelling a temporal behaviour. We will deal with transition systems only.

Furthermore, there also are various logics that allow the formalisation of temporal properties over transition systems. Emerson and Halpern's **CTL*** (cf. [6]), the full branching time logic, is not just one of them but probably the most appropriate one for expressing temporal properties. **BLTL**, the branching version of Pnueli's linear time logic **LTL** (cf. [13]) and Clarke and Emerson's branching time logic **CTL** (cf. [2]) for example can be found as genuine syntactic fragments of **CTL***. A lot of interesting properties, like "something holds infinitely often", cannot be expressed in **CTL** but in **CTL***. **BLTL** is capable of doing this, but cannot formalise the existence of a certain sequence of states in the system.

On the other hand, **CTL*** can be translated into Kozen's modal μ-calculus $\mathcal{L}_\mu$, introduced in [9], for which such model checking games already exist (cf. [16, 7]). However, the alternation depth of the resulting μ-calculus formulas is bounded by two (cf. [4]). Since model checking for **CTL*** is PSPACE-complete (cf. [3, 11]), whereas there exists a polynomial time algorithm solving the model checking problem for that fragment of $\mathcal{L}_\mu$, the translation procedure must enlarge the formulas or the transition systems exponentially, unless P=PSPACE.

Such translations are undesirable for the mentioned specification and verification process, perhaps for complexity reasons but also since a translation violates the *subformula property*: All formulas occurring in the game should be subformulas of the formalised property. This enables the user of a verification tool best to understand the diagnosis of the underlying system that is provided by an interactive game.

There is a fairly simple way of defining games for **CTL*** which follows exactly the semantics. I.e. whenever a path formula is reached the corresponding player names a whole path on which the formula is examined. However, it is easy to give examples in which the length of a shortest path to be chosen is at least as great as the size of the transition system. Moreover, since an algorithm might have to examine all the possible choices the players can take, the resulting complexity would be unacceptably high. Therefore we require paths in the transition system to be constructed stepwise throughout the game.

In [1] Bernholtz, Vardi and Wolper have shown how to solve the model checking problem for **CTL*** using *alternating automata* (cf. [12]). Their general approach is to translate the **CTL*** formula into a *hesitant alternating automaton over trees*. Forming the product of this with a Kripke structure results in an hesitant alternating automaton over words. The model checking problem is thus reduced to a non-emptiness check of that automaton. In [18] Barringer and Visser have shown how to do this check efficiently using games as well. Still their games rely on the construction of the alternating automata and therefore require knowledge of fairly advanced automata theory.

The model checking games of this paper can be viewed as alternating automata as well. Their winning conditions may correspond to a special Rabin acceptance condition on the automata side (cf. [14, 17]). However, using the games in this paper results in automata over infinite words immediately without taking the detour via satisfiability checking using alternating tree automata. It only requires an insight into the logic **CTL*** rather than another theory that is used to process formulas.

Section 2 recalls the syntax and semantics of **CTL*** and some of its fragments,

section 3 contains the definition of the model checking games and their correctness proof. Section 4 describes a game based algorithm and examines the complexity of the model checking problem for **CTL*** via games. Finally, optimisations of the games are considered in section 5 to customise the games for other branching time logics.

2 Syntax and Semantics

Let *Prop* be a set of propositional constants including *true* and *false*, which is closed under complementary propositions, i.e. $Prop = \{\mathtt{tt}, \mathtt{ff}, q_1, \overline{q_1}, \ldots\}$ where $\overline{\overline{q}} = q$ and $\overline{\mathtt{tt}} = \mathtt{ff}$. A *transition system* $\mathcal{T}$ is a triple (S, T, L) with (S, T) being a directed graph. $L : S \to 2^{Prop}$ labels the states, such that for all $s \in S$: $\mathtt{tt} \in L(s), \mathtt{ff} \notin L(s)$ and $q \in L(s)$ iff $\overline{q} \notin L(s)$. We assume that every state in the graph has at least one successor state.

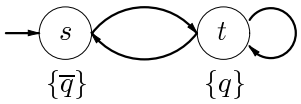

Figure 1: A transition system.

Definition 1 Formulas of temporal logics are built up from atomic propositions $q \in Prop$, the boolean connectives $\wedge$ and $\vee$, the temporal *next* (X), *until* (U), its dual *release* (R), and the path quantifiers A and E. The latter ones together with X are unary, all other constructs are binary. We will write Q for either A or E. The set of subformulas $Sub(\varphi)$ for a given φ is defined in the usual way, except that

$$\begin{array}{lcl} Sub(\varphi U \psi) & := & \{\varphi U \psi, X(\varphi U \psi), \varphi \wedge X(\varphi U \psi), \psi \vee (\varphi \wedge X(\varphi U \psi))\} \\ & & \cup Sub(\varphi) \cup Sub(\psi) \\ Sub(\varphi R \psi) & := & \{\varphi R \psi, X(\varphi R \psi), \varphi \vee X(\varphi R \psi), \psi \wedge (\varphi \vee X(\varphi R \psi))\} \\ & & \cup Sub(\varphi) \cup Sub(\psi) \end{array}$$

Not surprisingly, we set $Sub(\Phi) := \bigcup_{\varphi \in \Phi} Sub(\varphi)$ for a set Φ of formulas. For simplicity reasons we use the helpful abbreviations $F\psi := \mathtt{tt} U \psi$ and $G\psi := \mathtt{ff} R \psi$.

Definition 2 The semantics of temporal formulas is explained using full paths $\pi = s_0 s_1 \ldots s_n \ldots$ of a transition system $\mathcal{T}$. With $\pi^{(i)}$ we denote the suffix of π beginning with the state s_i. We assume the transition system to be fixed for the remainder of the paper and thus write $\pi \models \varphi$ instead of $\mathcal{T}, \pi \models \varphi$ whenever it is possible.

- $\pi \models q$ iff $q \in L(s_0)$
- $\pi \models \varphi \wedge \psi$ iff $\mathcal{T}, \pi \models \varphi$ and $\pi \models \psi$
- $\pi \models \varphi \vee \psi$ iff $\mathcal{T}, \pi \models \varphi$ or $\pi \models \psi$
- $\pi \models A\varphi$ iff for all paths $\sigma = s_0 \sigma' : \sigma \models \varphi$
- $\pi \models E\varphi$ iff there exists a path $\sigma = s_0 \sigma'$ and $\sigma \models \varphi$

- $\pi \models X\varphi$ iff $\pi^{(1)} \models \varphi$
- $\pi \models \varphi U \psi$ iff there exists $i \in \mathbb{N}$ s.t. $\pi^{(i)} \models \psi$ and for all $j < i : \pi^{(j)} \models \varphi$
- $\pi \models \varphi R \psi$ iff for all $i \in \mathbb{N} : \pi^{(i)} \models \psi$ or there exists a $j < i$ s.t. $\pi^{(j)} \models \varphi$

φ and ψ are *logically equivalent*, $\varphi \equiv \psi$, if for all transition systems $\mathcal{T}$ and paths π the following holds: $\mathcal{T}, \pi \models \varphi$ iff $\mathcal{T}, \pi \models \psi$. A temporal formula φ is called a *state formula* if $\varphi \equiv A\varphi$ holds. Hence we may write $s_0 \models \varphi$ instead of $\pi \models \varphi$ in case φ is a state formula. Formulas not being state formulas are called *path formulas*. They will still occur as subformulas of state formulas.

The temporal operators until and release can be characterised by their unfoldings $\varphi U \psi \equiv \psi \vee (\varphi \wedge X(\varphi U \psi))$ and $\varphi R \psi \equiv \psi \wedge (\varphi \vee X(\varphi R \psi))$ or as solutions to similar fixed point equations with until being a least fixed point and release a greatest.

With all operators being defined the particular temporal logics can be given by simple grammars.

Definition 3 The *pure branching time logic* **CTL*** enjoys no restriction on the use of the constructs at all.

$$\begin{array}{lcl} \varphi & ::= & A\psi \\ \psi & ::= & q \mid \psi \wedge \psi \mid \psi \vee \psi \mid X\psi \mid \psi U \psi \mid \psi R \psi \mid E\psi \mid A\psi \end{array}$$

Every **CTL*** formula begins with an A path quantifier to ensure that it is a state formula.[1]

Definition 4 The *branching time logic* **CTL** is characterised by the fact that every temporal operator is immediately preceded by a path quantifier.

$$\begin{array}{lcl} \varphi & ::= & q \mid \varphi \wedge \varphi \mid \varphi \vee \varphi \mid A\psi \mid E\psi \\ \psi & ::= & X\varphi \mid \varphi U \varphi \mid \varphi R \varphi \end{array}$$

Obviously, every **CTL** formula is a state formula.

Definition 5 **CTL**$^+$ allows boolean combinations of path formulas without nested temporal operators in addition to the features of **CTL**.

$$\begin{array}{lcl} \varphi & ::= & q \mid \varphi \wedge \varphi \mid \varphi \vee \varphi \mid A\psi \mid E\psi \\ \psi & ::= & \psi \wedge \psi \mid \psi \vee \psi \mid X\varphi \mid \varphi U \varphi \mid \varphi R \varphi \end{array}$$

Definition 6 The *fair branching time logic* **FCTL** allows path formulas to be interpreted over *fair* paths only. That is a path satisfying a fairness constraint which is a boolean combination of *infinitely* and *almost everywhere* operators over **CTL** formulas.

$$\begin{array}{lcl} \varphi & ::= & q \mid \varphi \wedge \varphi \mid \varphi \vee \varphi \mid A(\chi \rightarrow \psi) \mid E(\chi \wedge \psi) \\ \psi & ::= & X\varphi \mid \varphi U \varphi \mid \varphi R \varphi \\ \chi & ::= & \chi \wedge \chi \mid \chi \vee \chi \mid GF\varphi \mid FG\varphi \end{array}$$

where $\chi \rightarrow \psi := \neg\chi \vee \psi$. Negation can be driven inwards according to the next Lemma.

[1] This is not a restriction because of the equivalence $Q_1 Q_2 \varphi \equiv Q_2 \varphi$.

Lemma 7 **CTL***, **CTL**, **CTL**$^+$, and **FCTL** are closed under negation.

Proof Extend the syntax with a negation symbol $\neg$ whose semantics is given by $\pi \models \neg\varphi$ iff $\pi \not\models \varphi$. The propositional constants and boolean connectives are handled by the definition of a labelling function and deMorgan's laws. Furthermore, the equivalences $\neg A\varphi \equiv E\neg\varphi, \neg X\varphi \equiv X\neg\varphi$, and $\neg(\varphi U\psi) \equiv \neg\varphi R\neg\psi$ hold.

Definition 8 The *linear time logic* **LTL** is interpreted over paths of a transition system. Therefore it consists of temporal operators only. To enable the interpretation over transition systems, too, one universal path quantifier is added at the outermost position. The resulting sublogic of **CTL*** is called **BLTL**, the *branching version of* **LTL**.

$$\begin{array}{lcl} \varphi & ::= & A\psi \\ \psi & ::= & q \mid \psi \wedge \psi \mid \psi \vee \psi \mid X\psi \mid \psi U\psi \mid \psi R\psi \end{array}$$

3 Model Checking Games for CTL*

In order to introduce games we need two players, namely I and II.[2] If p is one of them then $\overline{p}$ denotes the other one. It is player II's task to show that a formula is satisfied whereas player I tries to show the converse.[3]

The set of *configurations* for a transition system $\mathcal{T}$ and a formula φ is $Conf(\mathcal{T}, \varphi) = \{\mathrm{I}, \mathrm{II}\} \times S \times Sub(\varphi) \times 2^{Sub(\varphi)}$. A configuration is written $p, s \vdash [\psi], \Phi$ where p is a player called the *path player*, $s \in S$, $\psi \in Sub(\varphi)$ and $\Phi \subseteq Sub(\varphi)$. In this case ψ is said to be *in focus*. We may also write $p, s \vdash \Phi$ if there is a $\psi \in \Phi$ in focus that does not need explicit mentioning.

A *play* between player I and player II is a sequence of configurations. There are nineteen rules of the form

$$\frac{p, s \vdash [\varphi], \Phi}{p', s' \vdash [\varphi'], \Phi'}\ p''$$

for transforming configurations. They are to be read as: *If the actual configuration is* $p, s \vdash [\varphi], \Phi$ *then player* p'' *has to perform a choice and the next configuration is* $p', s' \vdash [\varphi'], \Phi'$.

The *side formulas*, i.e. those that are not in focus, can be seen as an insurance for the path player's opponent to redo a move that she has done before. This is necessary because the path player is allowed to choose the path stepwise along which a formula is examined.

At each configuration the set of side formulas together with the formula in focus can be understood as a disjunction (resp. conjunction) of formulas in case the path player is player I (resp. II).

A play for a transition system $\mathcal{T}$ with starting state s and a formula φ begins with the configuration $\mathrm{I}, s \vdash [\varphi]$.[4] From then on, the play proceeds according to the following rules.

[2] In the following "he" will stand for player I whereas "she" will be a synonym for either player II or both.

[3] Therefore, they are often called *refuter* and *verifier*, or from a logical point of view $\forall$*belard* and $\exists$*loise*.

[4] Note that by the construction of **CTL*** formulas φ is of the form $A\psi$, thus player I being the path player in the beginning of a play.

Once the focus is on a quantified formula a new path has to be chosen. Thus, all current sideformulas do not matter anymore.

$$(1)\ \frac{p, s \vdash [A\varphi], \Phi}{\mathrm{I}, s \vdash [\varphi]} \qquad (2)\ \frac{p, s \vdash [E\varphi], \Phi}{\mathrm{II}, s \vdash [\varphi]}$$

An explicit state formula can also be discarded in case the focus player does not want to prove (resp. refute) it in the current state.

$$(3)\ \frac{p, s \vdash [\varphi], Q\psi, \Phi}{p, s \vdash [\varphi], \Phi}\ \overline{p} \qquad (4)\ \frac{p, s \vdash [\varphi], q, \Phi}{p, s \vdash [\varphi], \Phi}\ \overline{p}$$

The four rules for a boolean connective in focus are almost straightforward. Note that it is not necessary to keep both disjuncts for example if the path player is player II because assumingly she knows which path she is going to choose.

$$(5)\ \frac{\mathrm{I}, s \vdash [\varphi_0 \wedge \varphi_1], \Phi}{\mathrm{I}, s \vdash [\varphi_i], \Phi}\ \mathrm{I} \qquad (6)\ \frac{\mathrm{I}, s \vdash [\varphi_0 \vee \varphi_1], \Phi}{\mathrm{I}, s \vdash [\varphi_i], \varphi_{1-i}, \Phi}\ \mathrm{II}$$

$$(7)\ \frac{\mathrm{II}, s \vdash [\varphi_0 \vee \varphi_1], \Phi}{\mathrm{II}, s \vdash [\varphi_i], \Phi}\ \mathrm{II} \qquad (8)\ \frac{\mathrm{II}, s \vdash [\varphi_0 \wedge \varphi_1], \Phi}{\mathrm{II}, s \vdash [\varphi_i], \varphi_{1-i}, \Phi}\ \mathrm{I}$$

The temporal operators U and R simply are unfolded.

$$(9)\ \frac{p, s \vdash [\varphi U \psi], \Phi}{p, s \vdash [\psi \vee (\varphi \wedge X(\varphi U \psi))], \Phi} \qquad (10)\ \frac{p, s \vdash [\varphi R \psi], \Phi}{p, s \vdash [\psi \wedge (\varphi \vee X(\varphi R \psi))], \Phi}$$

One could easily imagine similar rules for the abbreviated formulas $F\psi$ and $G\psi$ in which they are unfolded to $\psi \vee XF\psi$ and $\psi \wedge XG\psi$ respectively.

Now, applying those rules might generate an X-formula in focus. Before a play can proceed with that all side formulas have to be brought into this form, too. The rules for that are very similar to the ones above.

$$(11)\ \frac{\mathrm{I}, s \vdash [X\psi], \varphi_0 \wedge \varphi_1, \Phi}{\mathrm{I}, s \vdash [X\psi], \varphi_i, \Phi}\ \mathrm{I} \qquad (12)\ \frac{\mathrm{I}, s \vdash [X\psi], \varphi_0 \vee \varphi_1, \Phi}{\mathrm{I}, s \vdash [X\psi], \varphi_0, \varphi_1, \Phi}$$

$$(13)\ \frac{\mathrm{II}, s \vdash [X\psi], \varphi_0 \vee \varphi_1, \Phi}{\mathrm{II}, s \vdash [X\psi], \varphi_i, \Phi}\ \mathrm{II} \qquad (14)\ \frac{\mathrm{II}, s \vdash [X\psi], \varphi_0 \wedge \varphi_1, \Phi}{\mathrm{II}, s \vdash [X\psi], \varphi_0, \varphi_1, \Phi}$$

$$(15)\ \frac{p, s \vdash [X\chi], \varphi U \psi, \Phi}{p, s \vdash [X\chi], \psi \vee (\varphi \wedge X(\varphi U \psi)), \Phi}$$

$$(16)\ \frac{p, s \vdash [X\chi], \varphi R \psi, \Phi}{p, s \vdash [X\chi], \psi \wedge (\varphi \vee X(\varphi R \psi)), \Phi}$$

Once a configuration is reached in which every formula begins with an X, it is possible to go over to the next state on the path currently being examined.

$$(17)\ \frac{p, s \vdash [X\varphi_0], X\varphi_1, \ldots, X\varphi_k}{p, t \vdash [\varphi_0], \varphi_1, \ldots, \varphi_k}\ p,\ \ s \to t$$

Finally, there is a special rule that enables the focus player to react appropriately to the path player's moves.

$$(18)\ \frac{p, s \vdash [\varphi], \psi, \Phi}{p, s \vdash [\psi], \varphi, \Phi}\ \overline{p}$$

A *move* in a play consists of two steps. First, the path player and the focus determines which of the rules (1) – (17) applies,[5] and hence which player takes the next choice. After that the path player's opponent has the chance to reset the focus using rule (18).

A play is finished after a full move if it has reached a configuration

1. $p, s \vdash [q], \Phi$, or
2. $C = \mathrm{II}, s \vdash [\varphi U \psi], \Phi$ (resp. $C = \mathrm{I}, s \vdash [\varphi R \psi], \Phi$) after the play already went through C and player $\overline{p}$ never applied rule (18) in between, or
3. $p, s \vdash [\varphi], \Phi$ for the second time possibly using rule (18) in between.

In the first case player II wins if $q \in L(s)$, otherwise player I wins. In the second case player I wins if the formula in focus is $\varphi U \psi$, and player II if it is $\varphi R \psi$. In the third case p wins.

The model checking *game* $\Gamma_{\mathbf{CTL}^*}(\mathcal{T}, s, \varphi)$ for a $\mathbf{CTL}^*$ formula φ and a transition system $\mathcal{T}$ with starting state s is the tree of all possible plays for $\mathcal{T}, s$ and φ. We say p *wins* or has a *winning strategy* for $\Gamma_{\mathbf{CTL}^*}(\mathcal{T}, s, \varphi)$ if she can force every play into a configuration that makes her win the play.

The *successful game tree* for the winner p of a game $\Gamma_{\mathbf{CTL}^*}(\mathcal{T}, s, \varphi)$ is a refined version of the tree of all possible plays in $\Gamma_{\mathbf{CTL}^*}(\mathcal{T}, s, \varphi)$. At every configuration C that gives p the choice all but one successor C' are eliminated, such that p still wins if she chooses C'. If $\overline{p}$ has the choice in C then all successors from the game graph are preserved in the successful game tree. Abusing notation we will identify $\Gamma_{\mathbf{CTL}^*}(\mathcal{T}, s, \varphi)$ with the successful game tree for the winner as well.

Example 9 To illustrate the game rules we give an example. Let $\mathcal{T}$ be the transition system of figure 1. The formula to be examined is $\varphi := E(\overline{q} U (Gq))$.[6] Obviously, $\mathcal{T}$ with starting state s satisfies φ. The tree showing a winning strategy for player II with the rule numbers annotated is given in figure 2. The dots indicate a branch of the game tree that occurs twice. We use the abbreviation $\psi := \overline{q} U (Gq)$. Player II wins the play of the leftmost branch because of winning condition three, and the one right beside it because of condition two.

3.1 Correctness

We will show that player II has a winning strategy for a game if and only if the transition system and the starting state model the formula. In order to do this we need a few technical lemmas.

[5] A situation in which two different rules are applicable is possible. However, the order in which they are used does not effect the outcome of the game.

[6] The expressed property is "There exists a path with a finite prefix and an infinite suffix. On the prefix q never holds, on the suffix it always does."

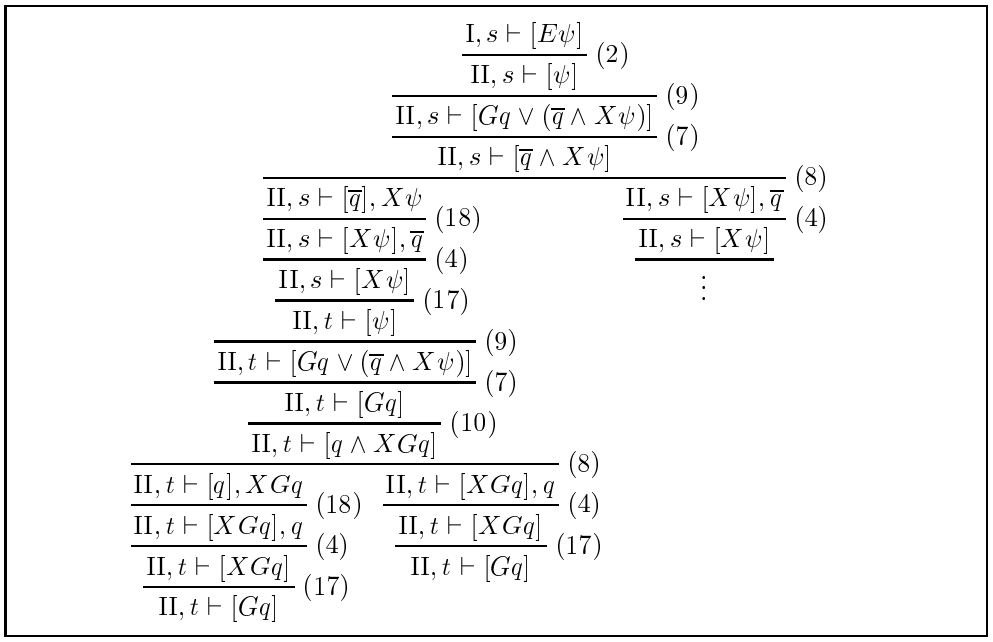

Figure 2: A successful game tree for player II.

Lemma 10 Let $C_0, \ldots, C, C_1, \ldots, C_k, C$ be a play with $C = p, s \vdash \Phi$. Then all intermediate configurations $C_1, \ldots, C_k$ are also of the form $p, s_i \vdash \Phi_i$ for $i = 1, \ldots, k$.

Proof For simplicity reasons we assume $p = \mathrm{I}$. Suppose there is an i with $C_i = \mathrm{II}, s_i \vdash \Phi_i$. Take the least such i. All formulas in Φ_i must have been subformulas of formulas in Φ. One of them must have been of the form $E\varphi$ which caused the path player to become player II with rule (2). From this follows $\Phi_i = \{\varphi\}$. As C is also a configuration following C_i all formulas in Φ must have been generated by φ only, in particular $E\varphi$. This would cause $E\varphi$ to be a genuine subformula of itself. The $p = \mathrm{II}$ case is dual.

Proposition 11 Every play has a uniquely determined winner.

Proof Every play is finite because the number of states of the transition system is finite, and so is the number of subformulas of a given φ. Therefore, the number of configurations is finite and every play will eventually reach a configuration that has been visited before and the third finishing condition will apply. The first or the second could apply beforehand.

If the play ended with an atomic proposition in focus then the winner is uniquely determined because $\{q, \overline{q}\} \subseteq L(s)$ is by definition excluded for each state s. If a configuration is visited twice then the path player, who is unique according to Lemma 10, wins. It may happen that a configuration $p, s \vdash \Phi$ with $\varphi U \psi, \varphi' R \psi' \in \Phi$ occurs twice, but only one of these formulas can stay in focus permanently. Hence, the winner is unique in this case, too.

Lemma 12 For every game one of the players has a winning strategy.

Proof Consider the tree of all possible plays in a given game. At each leaf one of the players has a winning strategy by doing nothing. Let C be a configuration with successors $C_1, \ldots, C_k$. By induction hypothesis, there is a winning strategy for either of the players in each game beginning with C_i. The branching in C can only be caused by one of the rules that require a choice to be made by, say, p. Now p also has a winning strategy for the game beginning in C if there exists an i such that p has a winning strategy in C_i, because she may choose to play on with C_i. If there is no such one, $\overline{p}$ has a winning strategy in C, because he will win no matter which C_i she chooses.

Corollary 13 Player II wins $\Gamma_{\mathbf{CTL}^*}(\mathcal{T}, s, \varphi)$ iff player I does not win $\Gamma_{\mathbf{CTL}^*}(\mathcal{T}, s, \varphi)$.

Proof The "only if" part is obvious. The "if" part follows directly from Lemma 12.

The next result reestablishes an observation from [8] in terms of games: $\mathbf{CTL}^*$ model checking can be polynomially reduced to **LTL** model checking.

Lemma 14 The game graph for a game $\Gamma_{\mathbf{CTL}^*}(\mathcal{T}, s, \varphi)$ can be partitioned into blocks. These blocks can be ordered, such that every play

- never leaves a block i into a block j with $j < i$, and
- finally stays in one block.

Proof This follows from Lemma 10 if one also considers changes from a path player p to p herself by using game rule (1) or (2). The order on the blocks can be found in a breadth-first-search that labels the reachable configurations with natural numbers, beginning with 1. A new number is assigned to a configuration whenever game rule (1) or (2) is applied. The second property follows from this and the finiteness of the game graph.

Lemma 15 Let Γ_1 and Γ_2 be two games beginning with $C_1 = p, s \vdash \Phi$ and $C_2 = p, s \vdash \Psi$. Assume that they both stay in one block only according to Lemma 14. Consider the game Γ_3 beginning with $C_3 = p, s \vdash \Phi \cup \Psi$.
a) If $p =$ I and player II wins Γ_1 or Γ_2 then she also wins Γ_3.
b) If $p =$ II and she wins Γ_1 and Γ_2 then she also wins Γ_3.

Proof a) Say she wins Γ_1. She will win Γ_3 by setting the focus as she would have done in Γ_1. Thus, she will also do the same moves. Since the set of side formulas in Γ_3 is larger than in Γ_1, rules (3), (4), (11) – (16) might have to be invoked at intermediate positions. However, the set is still finite such that only a finite number of new moves in Γ_3 can occur between two original moves from Γ_1. If she wins Γ_1 with winning condition one then she obviously does so in Γ_3, too. Assume she wins with condition two. The finiteness of the number of new side formulas from Γ_2 ensures that every play in Γ_3 performs a loop as well. Since formulas from Γ_2 do not occur in focus in this play the winner is the same as the one in Γ_1.

It is possible to create new branchings by using rule (11) for example. But the new plays only differ in the set of the side formulas which have no effect on the winner at all. Thus, every play in Γ_3 corresponds to a play in Γ_1 with the same winner.

b) Here, player I is in charge of the focus. Similar arguments as in the preceding case hold for the use of the rules (3), (4), (11) – (16) as well as for the loops in Γ_3. Player I can ignore side formulas, but he will lose because the plays correspond to similar plays in Γ_1 or Γ_2 where he would lose, too. Thus, his only chance is to reset the focus from a formula of, say, Γ_1 to a formula of Γ_2 before he loses like he would in Γ_1. Again, he will lose there as he would in Γ_2, or he resets the focus back to a formula from Γ_1 again. Since $|Sub(\Phi \cup \Psi)| < \infty$ he will eventually create a loop such that he used rule (18) on this loop. Thus, player II also wins every possible play in Γ_3.

Theorem 16 Player II wins $\Gamma_{\mathbf{CTL}^*}(\mathcal{T}, s, \varphi_0)$ iff $\mathcal{T}, s \models \varphi_0$.

Proof Because of Lemma 13 it is enough to prove one implication only. We will show completeness of the games by exhibiting a winning strategy for player II provided that φ_0 is satisfied by $\mathcal{T}$ and s. Soundness follows from completeness and duality (Corollary 13).

Because of Lemma 14 it suffices to consider games on formulas with one path quantifier only. Nested $Q\varphi$ formulas can be seen as completely new games and, hence, can be considered to be atomic propositions. An induction on the number of blocks finally proves the theorem for arbitrary $\mathbf{CTL}^*$ formulas. Therefore, we may assume the path player to stay the same throughout an entire game.

There are two distinguishable cases depending on the path quantifier of φ_0. First, let $\varphi_0 = A\varphi$. Thus, every configuration occurring in $\Gamma_{\mathbf{CTL}^*}(\mathcal{T}, s, \varphi_0)$ is of the form $\mathrm{I}, t \vdash \Phi$ where Φ is to be interpreted disjunctively, and player II has control over the focus. Furthermore, disjuncts are preserved and player I chooses at conjuncts.

In order to set the focus we let player II maintain a list L of all subformulas of φ_0, except atomic propositions. At the beginning L is ordered by decreasing size. At any stage in the play player II checks whether she can set the focus to a valid atomic proposition. If there is none she sets the focus to the earliest formula in the list that is present in the actual configuration. The formula that is in focus is always moved to the end of the list. She then takes her choices according to the following strategy. If the actual configuration is

- $\mathrm{I}, t \vdash [\chi \vee \psi], \Phi$ choose the disjunct that occurs first in L.
- $\mathrm{I}, t \vdash [\chi R \psi], \Phi$ choose $[\psi]$ or $[X(\chi R \psi)]$ after unfolding the R formula, depending on which of them was not discarded by player I.
- $\mathrm{I}, t \vdash [\chi U \psi], \Phi$ choose $[\psi]$ after the unfolding no matter which player I's choice is.
- $\mathrm{I}, t \vdash [q], \Phi$ and $q \notin L(t)$, set the focus to the earliest element in the list that is present in Φ.

These rules guarantee that in case of a repeat either an R formula stayed in focus or, if the focus has been changed, every other formula has been in focus, too. Thus player II avoids to lose with condition (3) although she could have won if she had set the focus cleverer.

It remains to show that this strategy guarantees her to win. Assume she does not. By Corollary 13 player I has a winning strategy and thus can force the play

into a winning position for himself. There are two possibilities. He can only win with winning condition (1) if the winning position is $\mathrm{I}, t \vdash [q]$, and there is no other formula that player II can set the focus to. But then $\varphi_0 = A(\bigwedge_i X^{k_i} q)$ for some i and k_i, and therefore $s \not\models \varphi_0$.

Player I's other possibility is to force a play like

$$\dfrac{\dfrac{\mathrm{I}, s \vdash [\varphi_0]}{\vdots}}{\dfrac{\mathrm{I}, t \vdash \Phi}{\vdots}}$$
$$\dfrac{}{\mathrm{I}, t \vdash \Phi}$$

where $\sigma_0 = s \ldots$ and $\sigma = t \ldots$ are the finite sequences of states occurring in the play such that player I has chosen the path $\pi = \sigma_0 \sigma t$. Moreover, there is a $[\varphi'] \in \Phi$ and $\varphi' = \chi R \psi$ only if player II has used the focus change rule between the two occurrences of the repeated configuration $C = \mathrm{I}, t \vdash \Phi$.

Since $s \models \varphi_0$ there is a $\alpha \in C$ s.t. $\sigma^\omega \models \alpha$. We show that player II is able to find this α and win before player I can win the play with winning condition (3). Let $\alpha = q$. Because q is true in t player II wins the play immediately.

In case $\alpha = \chi \wedge \psi$ the assumption shows that player I cannot win no matter which conjunct he chooses. The case $\alpha = X\psi$ is also straightforward since player II keeps the focus on α until game rule (17) is applied and the focus is automatically set to ψ.

If $\alpha = \chi \vee \psi$ she tries one of them, say χ. If $\sigma^\omega \not\models \chi$ then by hypothesis there will be an $i \in \mathbb{N}$ s.t. she has to reset the focus once the play has reached $(\sigma^\omega)^{(i)}$. But $\sigma^\omega \not\models \chi$ implies $\sigma^\omega \models \psi$, and there must be a $\psi' \in \mathit{Sub}(\psi)$ s.t. ψ' occurred later than χ in L and $(\sigma^\omega)^{(i)} \models \psi'$. Since χ is moved to the end of L player II will try ψ' before the play can perform a repeat on C.

If $\alpha = \chi U \psi$ then there is an $i \in \mathbb{N}$ s.t. $(\sigma^\omega)^{(i)} \models \psi$. According to her strategy player II will try to set the focus to ψ at every $(\sigma^\omega)^{(j)}$ with $j = 0, 1, \ldots$. If $(\sigma^\omega)^{(j)} \not\models \psi$ then, like in the former case, she will be forced to remove the focus from a formula that has been generated by ψ. But before she can reach ψ again she has to try $X(\chi U \psi)$. Therefore she will eventually find an i that has the ascribed property and, hence, disable the assumed repeat.

If $\alpha = \chi R \psi$ then there are two choices for player I to take. Either he chooses ψ. But $\sigma^\omega \models \chi R \psi$ implies $\sigma^\omega \models \psi$ and by induction hypothesis the assumed play is not possible. Or he chooses $\chi \vee X(\chi R \psi)$ which also holds on σ^ω. This case either reduced to the general $\vee$ case or player II sticks to the generated R formulas until the play reaches C again. But this contradicts the assumption that player I wins this play since the formula in focus has been $\chi R \psi$ and the focus has not been changed.

In the remaining case φ_0 begins with an existential path quantifier, i.e. $\varphi_0 = E\varphi$. Since $s \models \varphi_0$ there is a path π, s.t. $\pi \models \varphi$. Every configuration is of the form $\mathrm{II}, t \vdash \Phi$, and Φ is interpreted conjunctively, i.e. $\sigma \models \psi$ for all $\psi \in \Phi$ where $\sigma = \pi^{(i)} = t \ldots$ for some $i \in \mathbb{N}$. Suppose player I wins $\Gamma_{\mathbf{CTL}^*}(\mathcal{T}, s, \varphi_0)$. One possibility to do so is to exhibit a repeat on a $\chi U \psi$ in focus. Assume there is an $\alpha = X^l(\chi U \psi) \in \Phi$ for some $l \in \mathbb{N}$ and some configuration $\mathrm{II}, t \vdash \Phi$. Since $\sigma \models \alpha$ there is a $j \geq l$, s.t. $\sigma^{(j)} \models \psi$. Therefore player II can choose ψ once the play has reached the first state of $\sigma^{(j)}$.

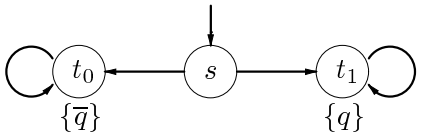

Figure 3: Another simple transition system.

This disables a repeat on $\chi U \psi$ in focus. However, $\chi U \psi$ might be regenerated by a superformula at a later position, but player I loses the play by winning condition (3) if he sets the focus to it at that point.

His other possibility to win is to reach a position $\mathrm{II}, t \vdash \Phi$ with $[q] \in \Phi$ and $q \notin L(t)$. But then $\sigma \not\models \bigwedge \Phi$ and therefore $\pi \not\models \varphi$.

Since this holds for all path choices by player II we conclude $s \not\models \varphi_0$ which contradicts the assumption.

3.2 The focus

In this section we show why a configuration in the model checking game needs to be a *set* of formulas and, moreover, why the focus on this set is needed, too.

Example 17 Consider the **CTL*** formula $\varphi := A(Xq \vee X\overline{q})$.[7] Obviously, φ is a tautology, so player I should not win any game on any transition system, in particular the $\mathcal{T}$ shown in figure 3.

However, if we require configurations to contain exactly one formula only, player II cannot win the game on $\mathcal{T}$ and φ anymore. The reason for this is that player II has to choose one of the disjuncts *before* player I chooses a transition from s to t_i, $i \in \{0, 1\}$. Clearly, if player II selected Xq for example he would choose t_0 and vice versa. Thus, configurations must be sets of formulas provided paths are chosen stepwise.

Example 18 The second example justifies the use of the extra focus structure on sets of formulas. Consider $\varphi := E(Fq \wedge GFq)$ and the two transition systems $\mathcal{T}_1 = (\{s\}, \{(s,s)\}, L_1)$ and $\mathcal{T}_2 = (\{s\}, \{(s,s)\}, L_2)$ with $L_1(s) = \{q\}$ and $L_2(s) = \{\overline{q}\}$. $\mathcal{T}_1, s \models \varphi$ but $\mathcal{T}_2, s \not\models \varphi$. However, without an additional structure like the focus on the set of formulas the games $\Gamma_{\mathbf{CTL}^*}(\mathcal{T}_1, s, \varphi)$ and $\Gamma_{\mathbf{CTL}^*}(\mathcal{T}_2, s, \varphi)$ would look like

$$\dfrac{\dfrac{\dfrac{\vdots}{\mathrm{II}, s \vdash Fq, XGFq}}{\vdots}}{\mathrm{II}, s \vdash Fq, XGFq} \qquad \dfrac{\dfrac{\dfrac{\vdots}{\mathrm{II}, s \vdash Fq, GFq}}{\vdots}}{\mathrm{II}, s \vdash Fq, GFq}$$

The difference between $\Gamma_{\mathbf{CTL}^*}(\mathcal{T}_1, s, \varphi)$, depicted on the left, and $\Gamma_{\mathbf{CTL}^*}(\mathcal{T}_2, s, \varphi)$ is the generation of Fq. In the first case it is generated from the $XGFq$ above, in the

[7] φ says that every path's next state is either labelled with q or $\overline{q}$.

second it regenerates itself. Hence, in that case player I can keep the focus on Fq and explicitly show this regeneration.

The proof of Theorem 16 shows that an algorithm which searches for a winning strategy for either of the players does not have to consider focus change moves after every step in a play. Indeed, it shows that focus change moves only need to be allowed

- to and away from an atomic or quantified subformula, or
- after the application of rule (17).

The first case is necessary because the winning condition (1) requires an atomic formula to be in focus. The second case is necessary because by using rule (17) the path player reveals a further step in the construction of a path while the focus player maintains a set of formulas among which there is one that is (not) fulfilled on this particular path and which depends on the path player's choice.

4 A Game Based Model Checking Algorithm

Because of Lemma 14 it is sufficient to consider formulas of the form $\varphi = A\psi$ or $\varphi = E\psi$ only where ψ does not contain any path quantifiers. For arbitrary formulas the algorithm $\mathbf{MC}_p$ can be recursively applied to every block of the game tree, where $p = \mathrm{I}$ if the path quantifier is a universal one. Otherwise $p = \mathrm{II}$.

The algorithm is shown below. It maintains three global variables, namely C_0 to find loops on paths, and *counter* and *max* to measure the length of a path. $\mathbf{MC}_p$ nondeterministically chooses successor configurations of the actual one to find a path on which the path player's opponent can keep a formula in focus such that she wins that play. Since the number of different configurations in one block is bounded by $|S| \cdot 2^{|\varphi|}$ the path player wins in case the length of a play exceeds this value.

```
global C_0 = I, s ⊢ [φ] ; counter = 0 ; max = |S| · 2^|φ|

MC_p(C) :=
    if C = C_0 then return p̄
    if counter ≥ max then return p
    choose C' ∈ next_moves(C)
    counter := counter + 1
    if focus_change(C, C') then C_0 := C'
    MC_p(C')
```

We can give an upper complexity bound of the model checking problem for $\mathbf{CTL}^*$ that matches the upper and lower bound from [3].

Theorem 19 The model checking problem for $\mathbf{CTL}^*$ is in PSPACE.

Proof An arbitrary $\mathbf{CTL}^*$ formula φ can contain at most $\frac{|\varphi|}{2}$ irredundant path quantifiers. Therefore, a game tree can contain at most $|S| \cdot \frac{|\varphi|}{2}$ blocks according to

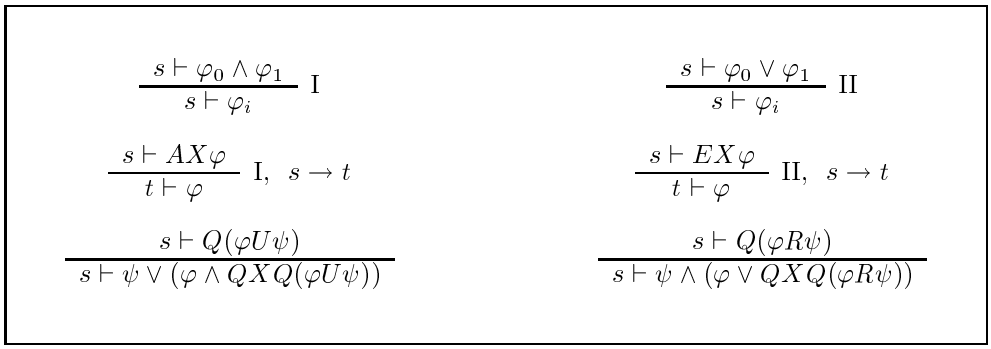

Figure 4: The rules for the **CTL** model checking games.

Lemma 14. Although $\mathbf{MC}_p$ might have to be invoked $|S| \cdot \frac{|\varphi|}{2}$ times, the space it needs can be reused for every call of the procedure.

Note that $\mathbf{MC}_p$ is end-recursive, thus it only needs to store the constant value *max*, the polynomially sized *counter*, and two configurations of size linear in φ.

By Savitch's Theorem (cf. [15]) the algorithm can be transformed into a deterministic one with a quadratic trade-off in the polynomial space complexity only.

The algorithm is local provided that the size of the transition system can be estimated without explicitly constructing it. This is possible if the transition system is given in a specification language like CCS (cf. [10]) for example.

5 Other Branching Time Logics

We examine how to optimise the model checking games for $\mathbf{CTL}^*$ in order to obtain model checking games for other branching time logics as well. $\Gamma_{\mathcal{L}}(\mathcal{T}, s, \varphi)$ is defined as in section 3 for a formula φ of a branching time logic $\mathcal{L}$.

5.1 Model Checking Games for CTL

A corollary of Lemma 14 says that in the **CTL** case no sets of formulas and hence no focus are needed. Since every temporal operator is immediately preceded by a path quantifier situations like the ones in examples 17 and 18 cannot occur. Moreover, whenever a temporal operator is handled the corresponding quantifier would cause all side formulas to be erased from a configuration anyway. Thus, the model checking game rules can be simplified vastly for the **CTL** case. They are given in figure 4. The set of configurations for the game on $\mathcal{T} = (S, T, L)$, $s \in S$ and φ is $Conf = S \times Sub(\varphi)$.[8]

A play is finished if it reaches a configuration

1. $C = s \vdash q$, or
2. $C = s \vdash Q(\chi U \psi)$ for the second time, or

[8] The definition of subformulas in the **CTL** case differs slightly from the one introduced in Def. 1 but can easily be seen in the game rules.

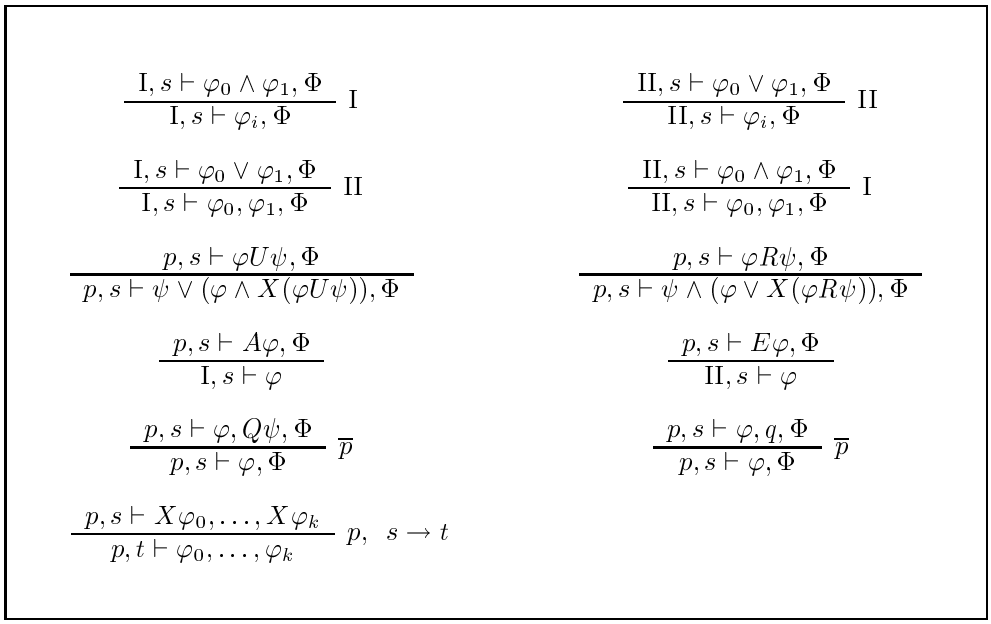

Figure 5: The model checking game rules for $\mathbf{CTL}^+$.

3. $C = s \vdash Q(\chi R \psi)$ for the second time.

In the first case player II wins if $q \in \lambda(s)$, otherwise player I wins. In the second case player I wins, and in the third case player II wins.

Theorem 20 Let $\mathcal{T} = (S, T, L), s \in S, \varphi \in \mathbf{CTL}$. $\mathcal{T}, s \models \varphi$ iff player II wins $\Gamma_{\mathbf{CTL}}(\mathcal{T}, s, \varphi)$.

Proof Every rule in a **CTL** game can be seen as a combination of rules of a $\mathbf{CTL}^*$ game, and the winning conditions are simply amended to these combined rules. Therefore correctness follows from Theorem 16.

5.2 Model Checking Games for $\mathbf{CTL}^+$

Although $\mathbf{CTL}^+$ has the same expressive power as **CTL** only (cf. [5, 19]), example 17 shows that the model checking games cannot be optimised for $\mathbf{CTL}^+$ like they can be for **CTL**. In particular, sets of formulas are needed. However, since the formula of example 18 that justifies the use of the focus is not in $\mathbf{CTL}^+$ the question whether a focus is needed for $\mathbf{CTL}^+$ games is reasonable to ask. $\mathbf{CTL}^+$ does not allow nested temporal operators, therefore the answer is no.

The game rules for the $\mathbf{CTL}^+$ model checking games are given in figure 5. Here, a play is finished if it reaches a configuration

1. $C = \mathrm{II}, s \vdash q, \Psi$, with $q \notin L(s)$, or else
2. $C = \mathrm{I}, s \vdash q, \Psi$, with $q \in L(s)$, or else

3. $C = \mathrm{II}, s \vdash \varphi U \psi, \Psi$ for the second time, or else

4. $C = \mathrm{I}, s \vdash \varphi R \psi, \Psi$ for the second time.

In the first and third case player I wins. So does player II in the second and fourth case. Note that the path player's opponent must be allowed to discard atomic propositions *before* one of the winning conditions can apply.

Theorem 21 Let $\mathcal{T} = (S, T, L), s \in S, \varphi \in \mathbf{CTL}^+$. $\mathcal{T}, s \models \varphi$ iff player II wins $\Gamma_{\mathbf{CTL}^+}(\mathcal{T}, s, \varphi)$.

Proof The game rules and winning conditions for $\mathbf{CTL}^+$ arise from the $\mathbf{CTL}^*$ games by removing the focus. Thus, it suffices to show that, whenever a play performs a loop, there is no ambiguity about the regeneration of U or R formulas. Assume a play like

$$\cfrac{\cfrac{\cfrac{\mathrm{I}, s \vdash \varphi}{\vdots}}{\mathrm{II}, s \vdash \chi U \psi, \varphi', \Psi}}{\vdots}$$
$$\mathrm{II}, s \vdash \chi U \psi, \varphi', \Psi$$

in which $\chi U \psi$ has been generated from φ' instead of from itself. Therefore, $\chi U \psi$ is a proper subformula of φ'. But this means that φ' contains a path quantified formula $Q\varphi''$ s.t. $\chi U \psi \in Sub(\varphi'')$. Since the rules for path quantifiers cause all other formulas to be erased φ' cannot occur a second time unless it was a proper subformula of itself. The $\chi R \psi$ case is dual.

5.3 Model Checking Games for FCTL and BLTL

Example 18 shows that already in the **FCTL** case a focus on sets of formulas is needed. This is not surprising since **FCTL** formulas may contain nested path operators of depth two.

Since **BLTL** formulas can contain arbitrary nestings of path operators together with boolean connectives the focus approach on sets of formulas is needed in that case, too. However, according to Lemma 14 the game graph for an **BLTL** formula consists of one block only. Therefore it is not necessary to memorise the path player explicitly. Rules (1) – (3) never apply, and in rule (17) it is always player I who chooses the next state from the transition system.

6 Conclusion

It was shown that model checking for branching time logics, in particular $\mathbf{CTL}^*$ can be done directly using games. In contrast to automata-theoretic approaches it is not necessary to take a detour via satisfiability checking first. Although the main advantage of games in logics is to provide a clear understanding of formulas and the properties they express, the games of this paper can be used to solve the model

checking problem for **CTL*** algorithmically in a way that matches the known lower and upper bounds.

The following table summarises the resources which are needed in the model checking games for the various branching time logics.

	Conf
CTL*	$\{\mathrm{I}, \mathrm{II}\} \times S \times Sub(\varphi) \times 2^{Sub(\varphi)}$
CTL	$S \times Sub(\varphi)$
CTL$^+$	$\{\mathrm{I}, \mathrm{II}\} \times S \times 2^{Sub(\varphi)}$
FCTL	$\{\mathrm{I}, \mathrm{II}\} \times S \times Sub(\varphi) \times 2^{Sub(\varphi)}$
BLTL	$S \times Sub(\varphi) \times 2^{Sub(\varphi)}$

References

[1] O. Bernholtz, M.Y. Vardi, and P. Wolper. An automata-theoretic approach to branching-time model checking. In D. L. Dill, editor, *Proc. 6th Int. Conf. on Computer Aided Verification, CAV'94*, volume 818 of *LNCS*, pages 142–155, Stanford, June 1994. Springer.

[2] E. M. Clarke and E. A. Emerson. Synthesis of synchronization skeletons for branching time temporal logic. In *Logics of Programs: Workshop*, volume 131 of *LNCS*, Yorktown Heights, New York, May 1981. Springer.

[3] E. M. Clarke, E. A. Emerson, and A. P. Sistla. Automatic verification of finite state concurrent systems using temporal logic specifications. In *Conf. Record of the 10th Annual ACM Symp. on Principles of Progr. Lang.*, pages 117–126. ACM, ACM, January 1983.

[4] M. Dam. CTL* and ECTL* as fragments of the modal μ-calculus. *TCS*, 126(1):77–96, April 1994.

[5] E. A. Emerson and J. Y. Halpern. Decision procedures and expressiveness in the temporal logic of branching time. *Journal of Computer and System Sciences*, 30:1–24, 1985.

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

[7] E. A. Emerson and C. S. Jutla. Tree automata, mu-calculus and determinacy. In IEEE, editor, *Proc. 32nd Annual Symp. on Foundations of Computer Science*, pages 368–377, San Juan, Puerto Rico, October 1991. IEEE Computer Society Press.

[8] E. A. Emerson and C.-L. Lei. Modalities for model checking: Branching time logic strikes back. *Science of Computer Programming*, 8(3):275–306, 1987.

[9] D. Kozen. Results on the propositional mu-calculus. *TCS*, 27:333–354, December 1983.

[10] R. Milner. A calculus of communicating systems. *LNCS*, 92, 1980.

[11] F. Moller and G. M. Birtwistle. *Logics for concurrency: structure versus automata*, volume 1043 of *LNCS*. Springer, New York, NY, USA, 1996.

[12] D. E. Muller and P. E. Schupp. Alternating automata on infinite trees. *TCS*, 54(2-3):267–276, October 1987.

[13] A. Pnueli. The temporal logic of programs. In *Proc. 18th IEEE Symp. on the Foundations of Computer Science, FOCS-77*, pages 46–57, Providence, Rhode Island, October 31–November 2 1977. IEEE, IEEE Computer Society Press.

[14] M. O. Rabin. Automata on infinite objects and church's problem. *Amer. Math. Soc. Providence, RI*, 1972.

[15] W. J. Savitch. Deterministic simulation of nondeterministic turing machines. In *ACM Symp. on Theory of Computing (STOC '69)*, pages 247–248, New York, May 1969. ACM Press.

[16] C. Stirling. Local model checking games. In I. Lee and S. A. Smolka, editors, *Proc. 6th Int. Conf. on Concurrency Theory, CONCUR'95*, volume 962 of *LNCS*, pages 1–11, Berlin, GER, August 1995. Springer.

[17] W. Thomas. Languages, automata and logic. In A. Salomaa and G. Rozenberg, editors, *Handbook of Formal Languages*, volume 3, Beyond Words. Springer, Berlin, 1997.

[18] W. Visser and H. Barringer. Practical CTL* model checking: Should SPIN be extended? *Int. J. on Software Tools for Technology Transfer*, 2(4):350–365, 2000.

[19] T. Wilke. CTL$^+$ is exponentially more succinct than CTL. *LNCS*, 1738:110–121, 1999.