Extending Fairness Expressibility of ECTL$^+$: A Tree-Style One-Pass Tableau Approach

Alexander Bolotov[1]
University of Westminster,
W1W 6UW, London, UK
A.Bolotov@westminster.ac.uk

Montserrat Hermo[2]
University of the Basque Country
P. Manuel de Lardizabal, 1, 20018-San Sebastián, Spain
montserrat.hermo@ehu.eus

Paqui Lucio[3]
University of the Basque Country
P. Manuel de Lardizabal, 1, 20018-San Sebastián, Spain
paqui.lucio@ehu.eus

Abstract

Temporal logic has become essential for various areas in computer science, most notably for the specification and verification of hardware and software systems. For the specification purposes rich temporal languages are required that, in particular, can express fairness constraints. For linear-time logics which deal with fairness in the linear-time setting, one-pass and two-pass tableau methods have been developed. In the repository of the CTL-type branching-time setting, the well-known logics ECTL and ECTL$^+$ were developed to explicitly deal with fairness. However, due to the syntactical restrictions, these logics can only express restricted versions of fairness. The logic CTL*, often considered as "the full branching-time logic" overcomes these restrictions on expressing fairness. However, this logic itself, is extremely challenging for the application of verification techniques, and the tableau technique, in particular. For example, there is no one-pass tableau construction for this logic, while it is known that one-pass tableau has an additional benefit enabling the formulation of dual sequent calculi that are often treated as more "natural" being more friendly for human understanding. Based on these two considerations, the following problem arises - are there logics that have richer expressiveness than ECTL$^+$ yet "simpler" than CTL* for which a one-pass tableau can be developed? In this paper we give a solution to this problem. We present a tree-style one-pass tableau for a sub-logic of CTL* that we call ECTL$^\#$, which is more expressive than ECTL$^+$ allowing the formulation of a new range of fairness constraints with "until" operator. The presentation of the tableau construction is accompanied by an algorithm for constructing a systematic tableau, for any given input of admissible branching-time formulae. We prove the termination, soundness and completeness of the method. As tree-shaped one-pass tableaux are well suited for the automation and are amenable for the implementation and for the formulation of sequent calculi, our results also open a prospect of relevant developments of the automation and implementation of the tableau method for ECTL$^\#$, and of a dual sequent calculi.

2012 ACM Subject Classification Theory of computation → Modal and temporal logics

[1] The author would like to thank the University of Westminster for supporting the sabbatical in 2017.
[2] This author has been partially supported by Spanish Projects TIN2013-46181-C2-2-R and TIN2017-86727-C2-2-R, and by the University of the Basque Country under Project LoRea GIU15/30.
[3] This author has been partially supported by Spanish Projects TIN2013-46181-C2-2-R and TIN2017-86727-C2-2-R, and by the University of the Basque Country under Project LoRea GIU15/30.

25th International Symposium on Temporal Representation and Reasoning (TIME 2018).
Editors: Natasha Alechina, Kjetil Nørvåg, and Wojciech Penczek; Article No. 5; pp. 5:1–5:22

Leibniz International Proceedings in Informatics
Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

Keywords and phrases Temporal logic, fairness, tableau, branching-time, one-pass tableau

Digital Object Identifier 10.4230/LIPIcs.TIME.2018.5

Acknowledgements The authors are grateful to Jose Gaintzarain for his contribution.

1 Introduction

Temporal logic has become essential for the specification and verification of hardware and software systems. For the specification of the reactive and distributed systems, or, most recently, autonomous systems, the modelling of the possibilities "branching" into the future is essential. Branching-time logics (BTL) give us an appropriate framework. Among important properties of these systems, so called fairness properties are important. In the standard formalisation of fairness, operators $\Diamond$ (eventually) and $\Box$ (always) have been used: $\mathsf{A}\Diamond\Box p$ – "p" is true along all computation paths except possibly their finite initial interval, where "A" is "for all paths" quantifier, and $\mathsf{E}\Box\Diamond p$ – "p" is true along a computation path at infinitely many states, where "E" is "there exists a path" quantifier.

For the branching-time setting, the most used class of formalisms are "CTL" (Computation Tree Logic) type logics. CTL itself requires every temporal operator to be preceded by a path quantifier, thus, cannot express fairness. ECTL (Extended CTL) [5] enables simple fairness constraints but not their Boolean combinations. ECTL$^+$ [6] further extends the expressiveness of ECTL allowing Boolean combinations of temporal operators and ECTL fairness constraints (but not permitting their nesting). Both ECTL and ECTL$^+$ extend the expressiveness of CTL in tackling fairness, instead of changing the semantics of the logic, as Fair CTL did [3]. The logic CTL*, often considered as "the full branching-time logic" overcomes these restrictions on expressing fairness. However, this logic is extremely challenging for the application of any known technique of automated reasoning. From another perspective, the literature on fairness constraints, even in the linear-time setting, lacks the analysis of their formulation with the $\mathcal{U}$ ("until") operator. To the best of our knowledge, there are only a few research papers that raise or discuss the problem. Among them are [10], which introduces the logic LCTL, providing an extension of liveness constraints by the "until" operator. However, LCTL belongs to 'Fair CTL-type' logics [7]. "Generalised liveness assumptions, which allow to express that the conclusion $f_2 \mathcal{U} f_3$ of a liveness assumption $\Box(f_1 \Rightarrow (f_2 \mathcal{U} f_3))$ has to be satisfied" are addressed in [1]. The $\mathcal{U}$ operator in the formulation of the fairness can also be found in [15] which considers the sequential composition of processes, providing the following example - the composition of processes P_1 and P_2 "behaves as $P1$ until its termination and then behaves as $P2$". Finally, [11] utilises restricted linear-time fairness constraints with $\mathcal{U}$ in the linear-time setting. We are not aware of any other analysis of fairness constraints in branching-time setting using the $\mathcal{U}$ operator and without restricting the underlying logic to be interpreted over the "fair" paths. We bridge this gap, presenting the logic ECTL$^\#$ (we use $\#$ to indicate some restrictions on concatenations of the modalities and their Boolean combinations). It is weaker than CTL* but extends ECTL$^+$ by allowing the combinations $\Box(A \mathcal{U} B)$ or $A \mathcal{U} \Box B$, referred to as modalities $\Box\mathcal{U}$ and $\mathcal{U}\Box$. This enables the formulation of stronger fairness constraints in the branching-time setting. The fairness constraint $\mathsf{A}(p \mathcal{U} \Box q)$ reads as "q is true along all paths of the computation except possibly their finite initial interval, where p is true", for example,"if a user of a system cannot repeat any old password and needs to change passwords from time to time, then the following property should hold: $\mathsf{A}((password = pw) \mathcal{U} \Box(password \neq pw))$ provided that pw is the current password". Table 1 places our logic in the hierarchy of BTL representing their

Table 1 Classification of CTL-type logics and their expressiveness.

$\mathcal{B}(\mathcal{U},\circ)$ (CTL) extensions	$\mathsf{E}(\Box\Diamond q)$	$\mathsf{E}(\Box\Diamond q \wedge \Box\Diamond r)$	$\mathsf{A}((p\,\mathcal{U}\,\Box q) \vee (s\,\mathcal{U}\,\Box\neg r))$	$\mathsf{A}\Diamond(\circ p \wedge \mathsf{E}\circ\neg p)$
$\mathcal{B}(\mathcal{U},\circ,\Box\Diamond)$ (ECTL)	$\surd$	X	X	X
$\mathcal{B}^{+}(\mathcal{U},\circ,\Box\Diamond)$ (ECTL^{+})	$\surd$	$\surd$	X	X
$\mathcal{B}^{+}(\mathcal{U},\circ,\mathcal{U}\Box)$ ($\mathsf{ECTL}^{\#}$)	$\surd$	$\surd$	$\surd$	X
$\mathcal{B}^{\star}(\mathcal{U},\circ)$ ($\mathsf{CTL}^{\star}$)	$\surd$	$\surd$	$\surd$	$\surd$

expressiveness: logics are classified by using "$\mathcal{B}$" for "Branching", followed by the set of only allowed modalities as parameters; $\mathcal{B}^{+}$ indicates admissible Boolean combinations of the modalities and $\mathcal{B}^{\star}$ reflects "no restrictions" in either concatenations of the modalities or Boolean combinations between them.[4] Thus, $\mathcal{B}(\mathcal{U},\circ)$ denotes the logic CTL. In this hierarchy $\mathsf{ECTL}^{\#}$ is $\mathcal{B}^{+}(\mathcal{U},\circ,\mathcal{U}\Box)$.

We present a tree-style one-pass tableau for this logic continuing the analogous developments in linear-time case [2, 8] and for CTL [2]. An indicative feature of this approach is context-based tableau technique. To the best of our knowledge, the context-based tableau has not been extended to more expressive BTL, though for them different other kinds of tableaux exist. In particular, [13] presents a tableau based decision procedure for $\mathsf{CTL}^{\star}$, which would definitely cover $\mathsf{ECTL}^{\#}$ as a sublogic of $\mathsf{CTL}^{\star}$. However, this tableau method is (unavoidably) complicated. For example, it utilises "global conditions on infinite branches" to be checked by the automata-theoretic approach. Though such complications may be well justified by the complexity of $\mathsf{CTL}^{\star}$, aiming at a weaker logic, we would benefit by reducing the complications to the minimum. Also, a distinctive feature of the tableau method in [13] is the control of loops, specifically, of so called "bad loops". While it looks necessary for this technique, we would like to avoid similar complications for a simpler logic. Moreover, due to the essential use of the notion of "context" (see §3) our tableau rules only produce "good loops". Tree-style one-pass tableaux (without additional procedures for checking meta-logical properties) have dual (cut-free) sequent calculi, see [8], enabling the construction of human-understandable proofs. In addition, these tableaux are well suited for the automation and are amenable for the implementation.[5] Our tableau is effectively an AND-OR tree where nodes are labelled by sets of state (see the definitions in §2) formulae. The difficult cases of $\mathsf{ECTL}^{\#}$ formulae appear due to the enriched syntax: disjunctions of formulae in the scope of the A quantifier and conjunctions of formulae in the scope of the E quantifier. To tackle these cases, in addition to $\alpha - \beta$ rules, that are standard to the tableaux, we use novel β^{+}-rules which use the context to force the eventualities to be fulfilled as soon as possible.

Outline of the paper. In §2 we describe the syntax and semantics of $\mathsf{ECTL}^{\#}$. The formulation of the tableau method is given in §3, where we define and explain tableau rules. A *systematic* tableau construction and relevant examples are introduced in §4. The correctness of our tableau method, its soundness, refutational completeness, and termination, are sketched in §5. In §6 we draw the conclusions and prospects of future work that the presented results open. Finally, the Appendix in §A collects the proofs of the main Propositions, Lemmas and Theorems.

[4] This notation goes back to [4], here we use its nice tuning by Nicolas Markey in [12].
[5] An excellent survey of the seminal tableau techniques for temporal logics can be found in [9].

2 Syntax and Semantics of ECTL$^\#$

In the language of ECTL$^\#$ we utilise classical connectives $(\neg, \wedge, \vee)$ [6], classically defined constants $\mathbf{F}$ ("false") and $\mathbf{T}$ ("true"), linear-time temporal operators $\Box$ (always), $\circ$ (next time), and $\mathcal{U}$ (until), and path quantifiers - A (on all future paths) and E (on some future path). Similarly to other BTL, we distinguish *state* (σ) and *path* (π) formulae, such that well formed formulae are state formulae.

▶ **Definition 1** (Syntax of ECTL$^\#$)**.** Let Prop be a fixed set of propositions, and let Lit be the set of literals, $Lit ::= \mathbf{F} \mid \mathbf{T} \mid \rho \mid \neg\rho$, where $\rho \in \mathsf{Prop}$. We inductively define the set of ECTL$^\#$-formulae, $\mathcal{F}_{\mathsf{Prop}}$, over Prop as follows:
$\sigma ::= Lit \mid \sigma_1 \wedge \sigma_2 \mid \sigma_1 \vee \sigma_2 \mid \mathsf{A}\pi \mid \mathsf{E}\pi$
$\pi ::= \pi_1 \wedge \pi_2 \mid \pi_1 \vee \pi_2 \mid \circ\sigma \mid \Box(\sigma \vee \Box\sigma) \mid \sigma\,\mathcal{U}\,(\sigma \wedge \Diamond\sigma) \mid \sigma\,\mathcal{U}\,(\Box\sigma) \mid \Box(\sigma\,\mathcal{U}\,\sigma)$
where σ means a state formula, π means a path formula and $\Diamond\sigma$ abbreviates $\mathbf{T}\,\mathcal{U}\,\sigma$.

Note that $\sigma_1\,\mathcal{U}\,\sigma_2$ and $\Box\sigma$ abbreviate $\sigma_1\,\mathcal{U}\,(\sigma_2 \wedge \Diamond\mathbf{T})$ and $\Box(\sigma \vee \Box\mathbf{F})$, respectively. Also, instead of using a minimal set of temporal operators, we use a richer syntax above to make the presentation of the tableau technique more transparent.

▶ **Definition 2** (Consistent Set of Formulae)**.** A set Σ of state formulae σ is *syntactically consistent* (we will write "consistent" further in the paper) abbreviated as $\Sigma_\top$ if $\mathbf{F} \notin \Sigma$ and $\{\sigma, \neg\sigma\} \not\subseteq \Sigma$ for any σ; otherwise, Σ is *inconsistent* abbreviated as $\Sigma_\bot$.

Formulae of ECTL$^\#$ are interpreted over labelled *Kripke structures*.

▶ **Definition 3** (Kripke structure $\mathcal{K} = (S, R, L)$)**.** A Kripke structure $\mathcal{K}$ is a triple of the form (S, R, L) such that S is a non-empty set of *states*, $R \subseteq S \times S$ is a total binary relation, called *the transition relation*, and $L : S \to 2^{\mathsf{Prop}}$ is a *labelling function*.

A *fullpath* x through a Kripke structure $\mathcal{K}$ is an infinite sequence of states $s_0, s_1, \ldots$ such that $(s_i, s_{i+1}) \in R$, for every $i \geq 0$. Given a path $x = s_0, s_1, \ldots, s_i, \ldots \quad (i \geq 0)$, we denote its state $s_i, 0 \leq i$, by $x(i)$ and its finite prefix by the sequence $x^{\leq i} = s_0, s_1, \ldots, s_i$. When path x is given, instead of $x(i)$ we will often write i, referring to i as "a state index of x". If x is a fullpath and y is a path such that $y(0) = x(i)$, for some $i > 0$, then the juxtaposition $x^{\leq i}y$ is a fullpath. Our Kripke structures are labelled directed graphs that correspond to Emerson's R-generable structures, i.e. the transition relation R is suffix, fusion and limit closed [4].

▶ **Definition 4** (Evaluation relation $\models$ for Kripke structures)**.** The relation $\mathcal{K}, x, i \models \varphi$ evaluates ECTL$^\#$ formula φ at the state index i of the given path x in the given structure $\mathcal{K} = (S, R, L)$ and is inductively defined as follows (we omit cases for Booleans, $\mathbf{F}$ and $\mathbf{T}$).

$\mathcal{K}, x, i \models \mathsf{A}\varphi$	iff	$\mathcal{K}, y, 0 \models \varphi$ holds for every path y such that $y(0) = x(i)$.
$\mathcal{K}, x, i \models \mathsf{E}\varphi$	iff	there exists a path y such that $y(0) = x(i)$ and $\mathcal{K}, y, 0 \models \varphi$.
$\mathcal{K}, x, i \models \circ\varphi$	iff	$\mathcal{K}, x, i+1 \models \varphi$.
$\mathcal{K}, x, i \models \Box\varphi$	iff	$\mathcal{K}, x, k \models \varphi$ holds for all $k \geq i$.
$\mathcal{K}, x, i \models \varphi_1\,\mathcal{U}\,\varphi_2$	iff	there exists $k \geq i$ such that $\mathcal{K}, x, k \models \varphi_2$ and $\mathcal{K}, x, j \models \varphi_1$ for all $j \in \{i, \ldots, k-1\}$.

[6] Since inputs to our tableau are supposed to be transformed into the negation normal form, see below.

Table 2 Difficult cases of temporal operators in the scope of path quantifiers.

Type of a difficult case	A-disjunctive formula	E-conjunctive formula
Example	$A(\circ q \vee \Box r)$	$E(\circ r \wedge q\,\mathcal{U}\,\Box\neg p)$
Our representation	$A(\circ q, \Box r)$	$E(\circ r, q\,\mathcal{U}\,\Box\neg p)$

We extend, in the standard way, the relation $\models$ to sets of formulae: given a set, Φ, of formulae, $\mathcal{K}, x, i \models \Phi$ iff $\mathcal{K}, x, i \models \varphi$, for all $\varphi \in \Phi$.

▶ **Definition 5** (Satisfiable Set of Formulae)**.** Given a set of formulae Φ, the set of its models, $\mathsf{Mod}(\Phi)$, is formed by all triples $(\mathcal{K}, x, i)$ such that $\mathcal{K}, x, i \models \Phi$. Then Φ is *satisfiable* ($\mathsf{Sat}(\Phi)$) if $\mathsf{Mod}(\Phi) \neq \emptyset$, otherwise Φ is *unsatisfiable* ($\mathsf{UnSat}(\Phi)$).

The sets Φ and Ψ are *equi-satisfiable* if the following holds: Φ is satisfiable iff Ψ is satisfiable. If $\mathsf{Mod}(\Phi) = \mathsf{Mod}(\Psi)$ then Φ and Ψ are equivalent denoted $\Phi \equiv \Psi$.

▶ **Definition 6** (Validity)**.** $\mathsf{ECTL}^{\#}$ formula σ is valid iff $\sigma \equiv \mathbf{T}$.

For a set of state formulae Σ, we sometimes write $\mathcal{K} \models \Sigma$ instead of $\mathcal{K}, x, 0 \models \Sigma$. For any $\mathcal{K}$, any $x \in \mathsf{fullpaths}(\mathcal{K})$ and any natural number i, the notation $\mathcal{K}\restriction x(i)$ denotes a Kripke structure with the set of states of $\mathcal{K}$ restricted to those that are R-reachable from $x(i)$. $\mathsf{CTL}^{\star}$, hence its sublogic $\mathsf{ECTL}^{\#}$ [14], has the *small model property*. Thus, we can consider *cyclic* $\mathsf{ECTL}^{\#}$-structures with fullpaths cyclic. Relevant concepts are defined below.

▶ **Definition 7** (Cyclic Sequence, Cyclic Path)**.** A finite sequence of states $z = s_0, s_1, \ldots, s_j$ is *cyclic* iff there exists s_i, $0 \leq i \leq j$ such that $(s_j, s_i) \in R$. The sequence $s_i, \ldots, s_j$ is a loop with a *cycling element* s_i. A *path* over z called *cyclic* is an infinite sequence where the loop $s_i, s_{i+1}, \ldots, s_j$ is repeated infinitely: $\mathsf{path}(z) = s_0, s_1, \ldots, s_{i-1}\langle s_i, s_{i+1}, \ldots, s_j\rangle^{\omega}$.

Cyclic paths do not have states after the "period", they are also called ultimately periodic.

▶ **Definition 8** (Cyclic Kripke structure)**.** A Kripke structure $\mathcal{K}$ is *cyclic* if every fullpath of $\mathcal{K}$ is a path over a cyclic sequence of states.

For $\mathsf{ECTL}^{\#}$, we identify the following difficult cases of the nesting and Boolean combinations of temporal operators in the scope of path quantifiers: A-disjunctive formula – disjunctions of temporal operators in the scope of A and E-conjunctive formula – conjunctions of temporal operators in the scope of E. For convenience, we will, respectively, write $\mathsf{A}(\pi_1, \ldots, \pi_n)$ and $\mathsf{E}(\pi_1, \ldots, \pi_n)$, where $n \geq 1$, and "," in the scope of A means $\vee$ and while in the scope of E it means $\wedge$. Examples given in Table 2 will be used to illustrate tableau, in Figure 2. Note that any A-formula (E-formula) σ can be transformed into an equivalent boolean combination of A-disjunctive formulae $\mathsf{A}(\pi_1, \ldots, \pi_n)$ (E-conjunctive formulae $\mathsf{E}(\pi_1, \ldots, \pi_n)$), such that every π_i $(1 \leq i \leq n)$ is of one of the following: $\circ\sigma$, $\sigma\,\mathcal{U}\,(\sigma \wedge \Diamond\sigma)$, $\sigma\,\mathcal{U}\,\Box\sigma$, $\Box(\sigma \vee \Box\sigma)$, and $\Box(\sigma\,\mathcal{U}\,\sigma)$, and σ stands for a state formula. For example, the formula $\mathsf{A}(((\circ q) \wedge (\Box\mathsf{E}\circ r)) \vee \circ p)$ is equivalent to $\mathsf{A}(\circ q, \circ p) \wedge \mathsf{A}(\Box\mathsf{E}\circ r, \circ p)$; and $\mathsf{E}(((\circ\mathsf{A}\circ r) \vee (q\,\mathcal{U}\,\Box\mathsf{E}\neg p)) \wedge \circ q)\}$ is equivalent to $\mathsf{E}(\circ\mathsf{A}\circ r) \vee \mathsf{E}(q\,\mathcal{U}\,\Box\mathsf{E}\neg p, \circ q)$. In what follows, Q abbreviates either of the path quantifiers. For a set of path formulae $\Pi = \{\pi_1, \ldots, \pi_n\}$, we write $\mathsf{Q}\Pi$ to denote the formula $\mathsf{Q}(\pi_1, \ldots, \pi_n)$, and $\mathsf{Q}\circ\Pi$ denotes $\mathsf{Q}(\circ\pi_1, \ldots, \circ\pi_n)$. An empty set of formulae Φ means $\mathbf{T}$ when Φ occurs in a conjunctive expression, while an empty Φ means $\mathbf{F}$ in a disjunctive expression. In particular, for empty Π, $\mathsf{A}\Pi = \mathbf{F}$ and $\mathsf{E}\Pi = \mathbf{T}$. We write Σ, σ to represent the set $\Sigma \cup \{\sigma\}$.

We assume that each $\sigma \in \mathcal{F}_{\mathsf{Prop}}$ is in its "negation normal form" called $\mathsf{nnf}(\sigma)$. The set of ECTL$^\#$-formulae is obviously closed under nnf: for any $\varphi \in \mathcal{F}_{\mathsf{Prop}}$, $\mathsf{nnf}(\neg\varphi) \in \mathcal{F}_{\mathsf{Prop}}$. Also, the negation of a state (path) formula is a state (path) formula. For example, $\mathsf{nnf}(\neg\mathsf{A}(p\,\mathcal{U}\,\Box q)) = \mathsf{E}((\Box\Diamond\neg q) \vee (\Diamond(\neg p \wedge \Diamond\neg q)))$. For simplicity, we will write $\neg\varphi$ instead of $\mathsf{nnf}(\neg\varphi)$, and for a finite set $\Delta = \{\varphi_1, \ldots, \varphi_n\}$, $\neg\Delta$ denotes the negation normal form of $\neg\bigwedge_{i=1}^{n}\varphi_i$.

3 The Tableau Method

3.1 Preliminaries

To make the subsequent sections more transparent we informally overview here the construction of the tableau. Recall that the initial set in a tableau is exclusively formed by state formulae (i.e. boolean combinations of literals and formulae of the form $\mathsf{Q}\Pi$).

▶ **Definition 9** (Tableau, Consistent Node, Closed branch)**.** A *tableau* for a set of state formulae Σ is a labelled tree T, where nodes are τ-labeled with sets of state formulae, such that the following two conditions hold:

(a) The root is labelled by the set Σ.

(b) Any other node m is labelled with sets of state formulae as the result of the application of one of the rules in Table 3, Table 4, Figure 1 and Table 5 to its parent node n. Given the applied rule is R, we term m an R-successor of n.

A node n of T is consistent, abbreviated as $n_\top$, if $\tau(n)$ is a consistent set of formulae (see Def. 2), else n is *inconsistent*, abbreviated as $n_\bot$. If for a branch b of T, there exists $n_\bot \in b$, then b is *closed* else b *open*.

A node of the tableau is labelled by a set of state formulae. To extend a node we apply one of the following three rules: α, β or β^+ rules. The first two types of rules are standard to the tableau, and are essentially based on the fixpoint characterisation of $\mathsf{Q}\Box$ and $\mathsf{Q}\,\mathcal{U}$ modalities, while β^+ rules are characteristic (and crucial!) for our construction. They tackle difficult cases of formulae in ECTL$^\#$, and are related to our dedicated account of the eventualities. Namely, we treat an eventuality as occurring in some *context*, which, in turn, is a collection of all state formulae (we will call this an outer context) or path formulae (we will call this an inner context). Subsequently, β^+ rules use the context to force eventualities to be fulfilled as soon as possible. $\alpha - \beta - \beta^+$ rules are applied to expand a node, generating its children labelled by the sets of state formulae. They apply repeatedly unless they produce an inconsistent node $n_\bot$, or we reach a node with the labels that already occurred within the path under consideration. In the former case the expansion of the given branch terminates with $n_\bot$ as a leaf. In the latter case, a repetitive node in a branch suggests that the formula under consideration is satisfied forever, so we change to select another eventuality (if any). Obviously, $n_\bot$ has an unsatifiable $\tau(n)$ and is a "deadlock" in the construction of a model. However, open branches do not necessarily give us a model. In particular, an open branch could be a prefix of a closed one. Later we introduce the notion of an expanded branch that enables the model construction. Once no more expansion rules are applicable to the given branch with the last node $n_\top$, the $\alpha - \beta$ rules ensure that $\tau(n) = \Sigma, \mathsf{A}\circ\Phi_1, \ldots, \mathsf{A}\circ\Phi_n, \mathsf{E}\circ\Psi_1, \ldots, \mathsf{E}\circ\Psi_m$ where Σ is a set of literals. As $\tau(n)$ only contains literals or formulae with the outer $\mathsf{Q}\circ$, this labelling is similar to a "pre-state" in the standard temporal tableau. Then the "next-state" rule applies to generate successors with the labels that are arguments of all $\mathsf{A}\circ$ modalities and the whole cycle of applying $\alpha - \beta - \beta^+$ and "next-state" rules is repeated until the tableau construction terminates. The nature of our rules ensures that the terminated tableau represents a model for the tableau input if

Table 3 ALPHA RULES. (σ_i, σ_j are state formulae, Π is a (possibly empty) set of path formulae.)

	α	S_α
$(\wedge)$	$\sigma_1 \wedge \sigma_2$	$\{\sigma_1, \sigma_2\}$
$(\mathsf{E}\sigma)$	$\mathsf{E}(\sigma_1, \ldots, \sigma_n, \Pi)$	$\{\sigma_1, \ldots, \sigma_n, \mathsf{E}\Pi\}$
$(\mathsf{E}\Box\mathcal{U})$	$\mathsf{E}(\Box(\sigma_1 \,\mathcal{U}\, \sigma_2), \Pi)$	$\{\mathsf{E}(\sigma_1 \,\mathcal{U}\, \sigma_2, \circ\Box(\sigma_1 \,\mathcal{U}\, \sigma_2), \Pi)\}$
$(\mathsf{A}\Box\mathcal{U})$	$\mathsf{A}(\Box(\sigma_1 \,\mathcal{U}\, \sigma_2), \Pi)$	$\{\mathsf{A}(\sigma_1 \,\mathcal{U}\, \sigma_2, \Pi), \mathsf{A}(\circ\Box(\sigma_1 \,\mathcal{U}\, \sigma_2), \Pi)\}$

Table 4 BETA RULES. (σ, σ_i are state formulae, Σ is a (possibly empty) set of state formulae, π_i is a path formula, Π is a (possibly empty) set of path formulae.)

β_Rule	β	k	$S_{\beta_i}(1 \leq i \leq k)$
$(\vee)$	$\sigma_1 \vee \sigma_2$	2	$S_{\beta_1} = \{\sigma_1\}$ $S_{\beta_2} = \{\sigma_2\}$
$(\mathsf{A}\sigma)$	$\mathsf{A}(\sigma_1, \ldots, \sigma_n, \Pi)$	$n+1$	$S_{\beta_1} = \{\sigma_1\}$ $\vdots$ $S_{\beta_n} = \{\sigma_n\}$ $S_{\beta_{n+1}} = \{\mathsf{A}\Pi\}$
$(\mathsf{E}\Box\sigma)$	$\mathsf{E}(\Box(\sigma_1 \vee \Box\sigma_2), \Pi)$	2	$S_{\beta_1} = \{\sigma_1, \mathsf{E}(\circ\Box(\sigma_1 \vee \Box\sigma_2), \Pi)\}$ $S_{\beta_2} = \{\neg\sigma_1, \sigma_2, \mathsf{E}(\circ\Box\sigma_2, \Pi)\}$
$(\mathsf{E}\mathcal{U}\,\sigma)$	$\mathsf{E}(\sigma_1 \,\mathcal{U}\, (\sigma_2 \wedge \Diamond\sigma_3), \Pi)$	2	$S_{\beta_1} = \{\sigma_2, \mathsf{E}(\Diamond\sigma_3, \Pi\}$ $S_{\beta_2} = \{\sigma_1, \mathsf{E}(\circ(\sigma_1 \,\mathcal{U}\, (\sigma_2 \wedge \Diamond\sigma_3)), \Pi)\}$
$(\mathsf{E}\mathcal{U}\,\Box)$	$\mathsf{E}(\sigma_1 \,\mathcal{U}\, \Box\sigma_2, \Pi)$	2	$S_{\beta_1} = \{\mathsf{E}(\Box\sigma_2, \Pi)\}$ $S_{\beta_2} = \{\sigma_1, \mathsf{E}(\circ(\sigma_1 \,\mathcal{U}\, \Box\sigma_2), \Pi)\}$
$(\mathsf{A}\Box\sigma)$	$\mathsf{A}(\Box(\sigma_1 \vee \Box\sigma_2), \Pi)$	3	$S_{\beta_1} = \{\sigma_1, \mathsf{A}(\circ\Box(\sigma_1 \vee \Box\sigma_2), \Pi)\}$ $S_{\beta_2} = \{\neg\sigma_1, \sigma_2, \mathsf{A}(\circ\Box\sigma_2, \Pi)\}$ $S_{\beta_3} = \{\mathsf{A}\Pi\}$
$(\mathsf{A}\mathcal{U}\,\sigma)$	$\mathsf{A}(\sigma_1 \,\mathcal{U}\, (\sigma_2 \wedge \Diamond\sigma_3), \Pi)$	3	$S_{\beta_1} = \{\sigma_2, \mathsf{A}(\Diamond\sigma_3, \Pi)\}$ $S_{\beta_2} = \{\sigma_1, \mathsf{A}(\circ(\sigma_1 \,\mathcal{U}\, (\sigma_2 \wedge \Diamond\sigma_3)), \Pi)\}$ $S_{\beta_3} = \{\mathsf{A}\Pi\}$
$(\mathsf{A}\mathcal{U}\,\Box)$	$\mathsf{A}(\sigma_1 \,\mathcal{U}\, \Box\sigma_2, \Pi)$	2	$S_{\beta_1} = \{\mathsf{A}(\Box\sigma_2, \Pi)\}$ $S_{\beta_2} = \{\sigma_1, \sigma_2, \mathsf{A}(\circ(\sigma_1 \,\mathcal{U}\, \Box\sigma_2), \Pi)\}$

all the leaves in a collection of branches, called a bunch, are consistent and all eventualities occurring in looping branches are fulfilled, otherwise, the tableau input is unsatisfiable. In addition to the tableau rules introduced in the rest of this Section, we also use simplifications rules that are given in the Appendix (Def. 42).

$$(\mathsf{Q}\bigcirc)\ \frac{\Sigma, \mathsf{A}\bigcirc\Phi_1, \ldots, \mathsf{A}\bigcirc\Phi_n, \mathsf{E}\bigcirc\Psi_1, \ldots, \mathsf{E}\bigcirc\Psi_m,}{\mathsf{A}\Phi_1, \ldots, \mathsf{A}\Phi_n, \mathsf{E}\Psi_1\ \&\ \ldots\ \&\ \mathsf{A}\Phi_1, \ldots, \mathsf{A}\Phi_n, \mathsf{E}\Psi_m}$$

Figure 1 Next-State Rule. (Set of literals Σ is possibly empty, $\Phi_i, \Psi_i \neq \emptyset$ are sets of formulae.)

3.2 Alpha and Beta Rules

The α- and β-rules are the most elementary rules. An application of an α-rule *enlarges* a branch with a node labelled by Σ, α, by a successor node labelled by Σ, S_α, where S_α is the set of formulae associated to α in Table 3. An α-rule has the following representation $\frac{\Sigma, \alpha}{\Sigma, S_\alpha}$. β-rules have the following representation $\frac{\Sigma, \beta}{\Sigma, S_{\beta_1} \mid \cdots \mid \Sigma, S_{\beta_k}}$. An application of a β-rule splits a branch containing a node with a set Σ, β, where β is one of the formulae of Table 4, in k new nodes each labelled by the corresponding Σ, S_{β_i}, see Table 4.

3.3 The Next-State Rule

The next-state rule ($\mathsf{Q}\bigcirc$) in Figure 1 is the only rule that splits branches in a *conjunctive way*: it produces m branches rooted by a node n labelled by a set $\mathsf{A}\Phi_1, \ldots, \mathsf{A}\Phi_n, \mathsf{E}\Psi_i$, for $i \in \{1, \ldots, m\}$. The generation of AND-successors of node n is represented by "&". If both n and m are zero, then the rule yields a unique new node labelled by the empty set. We assume that whenever m is zero and $n > 0$, there is a unique descendant labelled by $\mathsf{A}\Phi_1, \ldots, \mathsf{A}\Phi_n$.

▶ **Example 10.** Let n be a node such that $\tau(n) = \{a, \neg b, \mathsf{A}\bigcirc c, \mathsf{E}\bigcirc p, \mathsf{E}\bigcirc\neg p, \mathsf{A}\bigcirc\Box((\mathsf{E}\bigcirc p) \wedge (\mathsf{E}\bigcirc\neg p))\}$. Then only the next-state rule ($\mathsf{Q}\bigcirc$) can be applied to n generating the following AND-successors of n: $\{\mathsf{A}c, p, \mathsf{A}\Box((\mathsf{E}\bigcirc p) \wedge (\mathsf{E}\bigcirc\neg p))\}$ and $\{\mathsf{A}c, \neg p, \mathsf{A}\Box((\mathsf{E}\bigcirc p) \wedge (\mathsf{E}\bigcirc\neg p))\}$. Note that the formula $\mathsf{A}c$ requires the application of the β-rule ($\mathsf{A}\sigma$) to be reduced to c.

3.4 The Uniform Tableau

Here we present tableau with specific labels for leaves – *Uniform* sets of state formulae.

▶ **Definition 11** (Elementary Set of State Formulae)**.** A set of state formulae is *elementary* if and only if it is exclusively formed by literals and formulae of the form $Q\bigcirc\Pi$.

Repeatedly applying α-β-rules to consistent leafs, we get leaves labelled by elementary sets.

▶ **Proposition 12.** *Any set of state formulae has a tableau T such that all its leaves are labelled by elementary sets of state formulae.*

▶ **Definition 13** (Basic Path/State Formula, Uniform Set of Formulae)**.** Path formulae $\sigma_1\,\mathcal{U}\,(\sigma_2 \wedge \Diamond\sigma_3)$, $\sigma_1\,\mathcal{U}\,(\Box\sigma_2)$, $\Box(\sigma_1 \vee \Box\sigma_2)$, $\Box(\sigma_1\,\mathcal{U}\,\sigma_2)$ are *basic*. If Π is a set of basic path formulae then $Q\Pi$ is *basic*. A set of state formulae Σ is *uniform set* (US) iff Σ is only formed by literals and basic state formulae, and Σ contains at most one E-conjunctive formula.

▶ **Proposition 14.** *Any set of state formulae Σ has a tableau T s.t. all leaves are labelled by US of state formulae; all open branches contain exactly one application of* ($\mathsf{Q}\bigcirc$).

▶ **Definition 15** (Uniform Tableaux)**.** For any set Σ of state formulae, the tableau for Σ provided by Proposition 14 is denoted Uniform_Tableau(Σ).

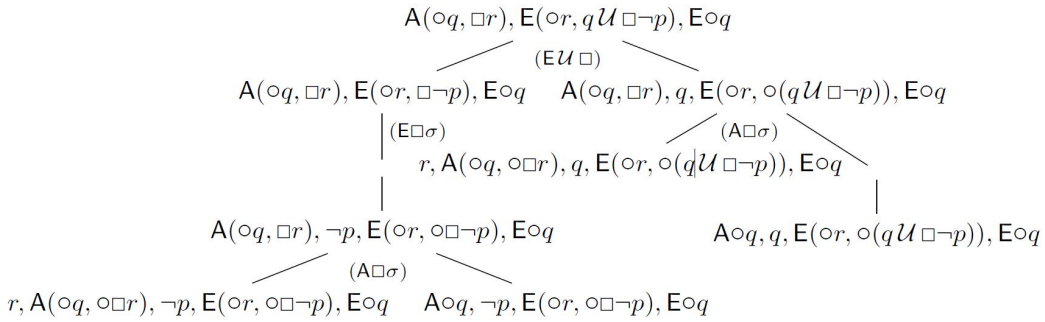

Figure 2 A tableau for $\mathsf{A}(\bigcirc q, \Box r), \mathsf{E}(\bigcirc r, q\,\mathcal{U}\,\Box\neg p)$; its leaves are elementary.

Table 5 Beta-Plus Rules. (Notation: σ, σ_i stand for state formulae, Σ is a (possibly empty) set of state formulae, Π is a (possibly empty) set of basic-path formulae.)

β^+-Rule	Σ, β	k	$S^+_{\Sigma,\beta_i}(1 \le i \le k)$
$(\mathsf{E}\,\mathcal{U}\,\sigma)^+$	$\Sigma, \mathsf{E}(\sigma_1\,\mathcal{U}\,(\sigma_2 \wedge \Diamond\sigma_3), \Pi)$	2	$S^+_{\Sigma,\beta_1} = \{\sigma_2, \mathsf{E}(\Diamond\sigma_3, \Pi)\}$ $S^+_{\Sigma,\beta_2} = \{\sigma_1, \mathsf{E}(\bigcirc((\sigma_1 \wedge \neg\Sigma)\,\mathcal{U}\,(\sigma_2 \wedge \Diamond\sigma_3)), \Pi)\}$
$(\mathsf{E}\,\mathcal{U}\,\Box)^+$	$\Sigma, \mathsf{E}(\sigma_1\,\mathcal{U}\,\Box\sigma_2, \Pi)$	2	$S^+_{\Sigma,\beta_1} = \{\mathsf{E}(\Box\sigma_2, \Pi)\}$ $S^+_{\Sigma,\beta_2} = \{\sigma_1, \mathsf{E}(\bigcirc((\sigma_1 \wedge \neg\Sigma)\,\mathcal{U}\,\Box\sigma_2), \Pi)\}$
$(\mathsf{A}\,\mathcal{U}\,\sigma)^+$	$\Sigma, \mathsf{A}(\sigma_1\,\mathcal{U}\,(\sigma_2 \wedge \Diamond\sigma_3), \Pi)$	3	$S^+_{\Sigma,\beta_1} = \{\sigma_2, \mathsf{A}(\Diamond\sigma_3, \Pi)\}$ $S^+_{\Sigma,\beta_2} = \{\sigma_1, \mathsf{A}(\bigcirc((\sigma_1 \wedge (\neg\Sigma \vee \varphi_\Pi))\,\mathcal{U}\,(\sigma_2 \wedge \Diamond\sigma_3)), \Pi)\}$ $S^+_{\Sigma,\beta_3} = \{\mathsf{A}\Pi\}$
$(\mathsf{A}\,\mathcal{U}\,\Box)^+$	$\Sigma, \mathsf{A}(\sigma_1\,\mathcal{U}\,\Box\sigma_2, \Pi)$	2	$S^+_{\Sigma,\beta_1} = \{\mathsf{A}(\Box\sigma_2, \Pi)\}$ $S^+_{\Sigma,\beta_2} = \{\sigma_1, \sigma_2, \mathsf{A}(\bigcirc((\sigma_1 \wedge \neg\Sigma \vee \varphi_\Pi \vee \sigma_2))\,\mathcal{U}\,\Box\sigma_2), \Pi)\}$

▶ **Example 16.** We construct a uniform tableau for the set $\{\mathsf{A}(\bigcirc q, \Box r), \mathsf{E}(\bigcirc r, q\,\mathcal{U}\,\Box\neg p), \mathsf{E}\bigcirc q)\}$. The first step gives the tableau in Figure 2. Subsequently, we enlarge each of the four branches by applying the rule $(\mathsf{Q}\bigcirc)$, obtaining the following eight leaves (left to right):

1. $\mathsf{A}(q, \Box r), \mathsf{E}(r, \Box\neg p)$
2. $\mathsf{A}(q, \Box r), \mathsf{E}q$
3. $\mathsf{A}q, \mathsf{E}(r, \Box\neg p)$
4. $\mathsf{A}q, \mathsf{E}q$
5. $\mathsf{A}(q, \Box r), \mathsf{E}(r, q\,\mathcal{U}\,\Box\neg p)$
6. $\mathsf{A}(q, \Box r), \mathsf{E}q$
7. $\mathsf{A}q, \mathsf{E}(r, q\,\mathcal{U}\,\Box\neg p)$
8. $\mathsf{A}q, \mathsf{E}q$

Then by rules $(\mathsf{A}\sigma)$ and $(\mathsf{E}\sigma)$: the first branch is split into $q, r, \mathsf{E}\Box\neg p$ and $\mathsf{A}\Box r, r, \mathsf{E}\Box\neg p$; the second into q and $\mathsf{A}\Box r, q$; the third yields only a child $q, r, \mathsf{E}\Box\neg p$; the fourth and the eighth yield only q; the fifth is split into two nodes $q, r, \mathsf{E}(q\,\mathcal{U}\,\Box\neg p)$ and $\mathsf{A}\Box r, r, \mathsf{E}(q\,\mathcal{U}\,\Box\neg p)$; the sixth into q and $\mathsf{A}\Box r, q$; and the seventh yields the unique child $q, r, \mathsf{E}(q\,\mathcal{U}\,\Box\neg p)$.

3.5 The Beta-plus Rules

We extend tableau rules with so called β^+-rules (Table 5), that, similarly to β-rules, split a branch into two or three branches. They apply to a set Σ, β, where β has the form $\mathsf{Q}(\pi, \Pi)$ and Π is a set of basic-path formulae and π is an $\mathcal{U}$-formula. The β^+ rules are the only rules that make use of the so-called context for forcing the eventualities to be satisfied as soon as possible. The context is given by the sets Σ containing state formulae and Π containing path formulae. We name Σ the outer context and Π the inner context. The outer context is used by all the β^+rules. The inner context is only needed to deal with formulae $\mathsf{A}\Pi$. Note that each β^+-rule, when applied to a formula of the form $\mathsf{Q}(\sigma_1\,\mathcal{U}\,\varphi, \Pi)$ –where φ could

be $\sigma_2 \wedge \Diamond\sigma_3$ or $\Box\sigma_2$– generates one or more successors that contain a formula of the form $\mathsf{Q}(\circ((\sigma_1 \wedge \sigma)\,\mathcal{U}\,\varphi), \Pi)$ where σ depends on both the inner and the outer context, and is defined based on whether Q is E or A. We call $(\sigma_1 \wedge \sigma)\,\mathcal{U}\,\varphi$ the *next-step variant* of $\sigma\,\mathcal{U}\,\varphi$.

▶ **Definition 17** (Formula φ_Π for β^+-rules)**.** Let Π be a set of basic path formulae. We define the formula φ_Π to be the following disjunction of state formulae:[7]

$$\bigvee_{\Box(\sigma_1 \vee \Box\sigma_2) \in \Pi} (\sigma_1 \vee \sigma_2) \;\vee\; \bigvee_{\sigma_1\,\mathcal{U}\,\Box\sigma_2 \in \Pi} \sigma_2 \;\vee\; \bigvee_{\Box(\sigma_1\,\mathcal{U}\,\sigma_2) \in \Pi} \mathsf{E}(\Diamond\sigma_2).$$

The proof of Proposition 29 and the role that this proposition plays in the proof of Lemma 30 point out some hints about the relation of the formula φ_Π (in Definition 17) with a limit path, which is a model for the formula $\mathsf{E}(\neg\pi_1, \ldots, \neg\pi_n)$, but is not a model for $\mathsf{A}(\pi_1, \ldots, \pi_n)$. The following example aims to provide some intuition on the role of φ_Π from the constructive view, i.e when we construct a tableau for a formula $\mathsf{A}(\pi_1, \ldots, \pi_n)$.

▶ **Example 18.** Consider the application of $(\mathsf{A}\,\mathcal{U}\,\sigma)^+$ rule to the formula $\mathsf{A}(a\,\mathcal{U}\,b, \Pi)$, where $\Pi = \{\Box c, r\,\mathcal{U}\,\Box s, \Box(p\,\mathcal{U}\,q)\}$ and $a, b, c, p, q, r, s \in \mathsf{Prop}$. The outer context, Σ, is empty and the inner context is Π. Then $\neg\Sigma = \mathbf{F}$ and $\varphi_\Pi = c \vee s \vee \mathsf{E}\Diamond q$. Hence, the second child, namely $S^+_{\beta_2}$, raised by the application of $(\mathsf{A}\,\mathcal{U}\,\sigma)^+$ is labelled by $\{a, \mathsf{A}(\circ((a \wedge (c \vee s \vee \mathsf{E}\Diamond q))\,\mathcal{U}\,b), \Pi)\}$. Then applying the rules $(\mathsf{A}\Box\sigma)$ to $\Box c$, $(\mathsf{A}\,\mathcal{U}\,\Box)$ to $r\,\mathcal{U}\,\Box s$, and $(\mathsf{A}\Box\,\mathcal{U}\,)$ (followed by $(\mathsf{A}\,\mathcal{U}\,\sigma)$) to $\Box(p\,\mathcal{U}\,q)$ we get, in one of the branches, a node n labelled by the set:

$$\{a, c, r, s, p, \mathsf{A}(\circ((a \wedge (c \vee s \vee \mathsf{E}\Diamond q))\,\mathcal{U}\,b), \circ\Box c, \circ(r\,\mathcal{U}\,\Box s), \circ\Box(p\,\mathcal{U}\,q))\} \tag{1}$$

Then, by rule $(\mathsf{Q}\circ)$, $\tau(n_1) = \{\mathsf{A}((a \wedge (c \vee s \vee \mathsf{E}\Diamond q))\,\mathcal{U}\,b, \Box c, r\,\mathcal{U}\,\Box s, \Box(p\,\mathcal{U}\,q))\}$, where n_1 is the unique child of n. Now, repeating the previous steps, we get a node m labelled by

$$\{a \wedge (c \vee s \vee \mathsf{E}\Diamond q), c, r, s, p, \mathsf{A}(\circ((a \wedge (c \vee s \vee \mathsf{E}\Diamond q))\,\mathcal{U}\,b), \circ\Box c, \circ(r\,\mathcal{U}\,\Box s), \circ\Box(p\,\mathcal{U}\,q))\}$$

Using the rules $(\wedge)$ and $(\vee)$, we get three children of m. Two of them are labelled by the set (1). Hence, by rule $(\mathsf{Q}\circ)$, we get a cycle to node n_1. It is worth noting that this branch represents a model where the initial A-disjunctive formula $\mathsf{A}(a\,\mathcal{U}\,b, \Box c, r\,\mathcal{U}\,\Box s, \Box(p\,\mathcal{U}\,q))$ is satisfied because both $\Box c$ and $r\,\mathcal{U}\,\Box s$ are satisfied. The third node is labelled by

$$\{a, \mathsf{E}\Diamond q, c, r, s, p, \mathsf{A}(\circ((a \wedge (c \vee s \vee \mathsf{E}\Diamond q))\,\mathcal{U}\,b), \circ\Box c, \circ(r\,\mathcal{U}\,\Box s), \circ\Box(p\,\mathcal{U}\,q))\}$$

Therefore, one of its children, due to the rule $(\mathsf{E}\,\mathcal{U}\,\sigma)$, is labelled by

$$\{a, q, c, r, s, p, \mathsf{A}(\circ((a \wedge (c \vee s \vee \mathsf{E}\Diamond q))\,\mathcal{U}\,b), \circ\Box c, \circ(r\,\mathcal{U}\,\Box s), \circ\Box(p\,\mathcal{U}\,q))\}$$

and after applying $(\mathsf{Q}\circ)$ we get a node labelled as n_1 is, obtaining a cycle. This cycling branch represents a model for the initial A-disjunctive formula $\mathsf{A}(a\,\mathcal{U}\,b, \Box c, r\,\mathcal{U}\,\Box s, \Box(p\,\mathcal{U}\,q))$ because $\Box(p\,\mathcal{U}\,q)$ is satisfied. Here, for simplicity, we consider an empty outer context. However, note that $\neg\Sigma$ is also a disjunction of state formulae. For non-empty Σ, the application of the rule $(\vee)$ would generate a child for each disjunct in $\neg\Sigma$. Each of these children represents a trial to satisfy the formula $a\,\mathcal{U}\,b$ (in the initial A-disjunctive formula) as soon as possible.

[7] If the disjunction is empty, then $\varphi_\Pi = \mathbf{F}$.

Algorithm 1 Systematic Tableau Construction.

procedure SYSTEMATIC_TABLEAU(Σ_0) ▷ where Σ_0: set of state formulae
  if Σ_0 is not uniform **then** $T :=$ Uniform_Tableau(Σ_0)
  while T has at least one non-terminal leaf **do**
    ▷ Invariant: Any non-terminal leaf of T is labelled by a US
    Choose any leaf ℓ in T such that $\tau(\ell)$ is not terminal
    Let $\Sigma = \tau(\ell)$ ▷ Σ is uniform
    if there are not eventualities in ℓ **then** $T := T[\ell \leftarrow$Uniform_Tableau(Σ)$]$
    else
      Eventuality_Selection(Σ)
      Apply_β^+-rule(Σ)
      Let $k \in \{2, 3\}$ the number of new leaves
      Let $\ell_1, \ldots, \ell_k$ be the new leaves and let $\Sigma_1, \ldots, \Sigma_k$ be their respective labels
      for i = 1 .. k **do**
        if ℓ_i is non-terminal and Σ_i is not uniform **then**
          $T := T[\ell_i \leftarrow$Uniform_Tableau(Σ_i)$]$
  return T

4 Systematic Tableau Construction

Here we define an algorithm, $\mathcal{A}^{sys}$, that constructs a *systematic tableau* and illustrate its performance. Recall that due to the rule (Q○), any open tableau should have a collection of open branches including all the (Q○)-successors of any node labelled by an elementary set of formulae. These collections of branches are called *bunches*. Any open bunch of the systematic tableau, constructed by $\mathcal{A}^{sys}$, gives us of a model for the initial set of formulae. $\mathcal{A}^{sys}$ constructs an *expanded* tableau (see Definition 34) for the given input. $\mathcal{A}^{sys}$ applied to the input Σ_0, denoted as $\mathcal{A}^{sys}(\Sigma_0)$, returns a systematic tableau $\mathcal{A}^{sys}_{\Sigma_0}$. Intuitively, expanded means "complete" in the sense that any possible rule has been already applied at every node. Though the best way to implement this algorithm is a depth-first construction, for the sake of clarity we prefer to formulate it as a breadth-first construction of a collection of subtrees. The procedure Uniform_Tableau in Algorithm 1 was introduced in Definition 15 along with the notion of a US of state formulae. The notation $T_1[\ell \leftarrow T_2]$ stands for the tableau T_1 where the leaf ℓ is substituted by the tableau T_2. In particular, $T[\ell \leftarrow$Uniform_Tableau(Σ)$]$ is the tableau T where the leaf ℓ is substituted by the Uniform_Tableau(Σ).

To introduce the procedures Eventuality_Selection and Apply_β^+-rule and related concepts of *terminal* node and *eventuality-covered* branch, let $\pi_{\mathcal{U}}$ denote a basic path formula that contains the operator $\mathcal{U}$, i.e. $\pi_{\mathcal{U}}$ is either $\sigma_1 \mathcal{U} (\sigma_2 \wedge \Diamond\sigma_3)$ or $\sigma_1 \mathcal{U} \Box\sigma_2$ or $\Box(\sigma_1 \mathcal{U} \sigma_2)$. We call these formulae *eventualities*. Consequently, the notation $\mathsf{Q}(\pi_{\mathcal{U}}, \Pi)$ stands for a formula that contains at least one eventuality.

Eventuality_Selection selects a state formula $\mathsf{Q}(\pi_{\mathcal{U}}, \Pi)$ (if there is one) and marks the eventuality $\pi_{\mathcal{U}}$ while Apply_β^+-rule(Σ) applies the corresponding rule (or pair of rules) depending on the selected eventuality:

- If $\pi_{\mathcal{U}} = \sigma_1 \mathcal{U} (\sigma_2 \wedge \Diamond\sigma_3)$ is the marked eventuality, then apply $(\mathsf{Q}\mathcal{U}\sigma)^+$
- If $\pi_{\mathcal{U}} = \sigma_1 \mathcal{U} \Box\sigma_2$ is the marked eventuality, then apply the respective rule $(\mathsf{Q}\mathcal{U}\Box)^+$
- If $\pi_{\mathcal{U}} = \Box(\sigma_1 \mathcal{U} \sigma_2)$ is the marked eventuality, then apply first the rule $(\mathsf{Q}\Box\mathcal{U})$ and then the rule $(\mathsf{Q}\mathcal{U}\sigma)^+$ with the selected eventuality $\sigma_1 \mathcal{U} \sigma_2$.

Each application of a β^+-rule introduces a next-step variant of the marked eventuality.

The call Eventuality_Selection(Σ) keeps the selection of $\mathsf{Q}\Pi \in \Sigma$ which contains a next-step variant of the previously marked eventuality, whenever the leaf $\ell, (\Sigma = \tau(\ell))$ is not a loop-node. If ℓ is a loop-node, then a new selection should be made, if possible. If the branch is already eventuality-covered and ℓ is a loop-node, ℓ is the leaf of an expanded open branch (see Definition 34). When making the selection, priorities are guided by Definition 19 - the idea is that the tableau branches represent paths in possible models. Since any path in a model is cyclic, it has a possibly empty initial sequence of states followed by a looping-sequence. The highest priority formulae are those that cannot produce a loop. The formulae that are not of highest priority can be potentially-cycling formulae or cycling. Once all the highest priority formulae have been selected, we only have cycling and potentially-cycling formulae. Now the objective is to have a sequence of loop-nodes.

▶ **Definition 19** (Priorities for Eventuality Selection)**.** The formulae of the *highest priority* for Eventuality_Selection are the formulae of the form:

- $\mathsf{A}\Pi$ where Π is exclusively formed by formulae of the form $\sigma_1 \,\mathcal{U}\, (\sigma_2 \wedge \Diamond\sigma_3)$, and
- $\mathsf{E}\Pi$ where Π contains at least one eventuality. Eventualities of the form $\sigma_1 \,\mathcal{U}\, (\sigma_2 \wedge \Diamond\sigma_3)$ and $\sigma_1 \,\mathcal{U}\, \Box\sigma_2$ have also the maximal priority to be marked in the selected $\mathsf{E}\Pi$.

▶ **Definition 20** (Cycling Formula)**.** A formula is *cycling* if it is of the form $\mathsf{Q}\Pi$ where Π only contains formulae of the form: $\Box(\sigma_1 \vee \Box\sigma_2)$ and $\Box(\sigma_1 \,\mathcal{U}\, \sigma_2)$.

For any E-conjunctive formula σ, σ is not highest priority if and only if σ is a cycling formula. For A-disjunctive formulae it is different - we also have potentially-cycling formulae.

▶ **Definition 21** (Potentially-Cycling Formula)**.** A formula is *potentially-cycling* if it is of the form $A\Pi$ where Π contains at least one formula of the form $\sigma_1 \,\mathcal{U}\, (\sigma_2 \wedge \Diamond\sigma_3)$ or $\sigma_1 \,\mathcal{U}\, \Box\sigma_2$ and also contains at least one formula of the form $\Box(\sigma_1 \vee \sigma_2)$ or $\Box(\sigma_1 \,\mathcal{U}\, \sigma_2)$.

▶ **Definition 22** (Loop-node)**.** Given b is a branch of T and $n_i \in b$ $(0 \leq i)$, n_i is a *loop-node* if there exists $n_j \in b$ $(0 \leq j < i)$ and $\tau(n_i) = \tau(n_j)$; n_j is called a *companion node* of n_i.

▶ **Definition 23** (Eventuality-covered Branch)**.** A branch $b = n_0, n_1, ..., n_i$ of T is *eventuality-covered* if n_i is a loop-node, with a companion node n_j $(0 \leq j < i)$, both labelled by a US $\Sigma = \mathsf{Q}_1\Pi_1, \ldots, \mathsf{Q}_m\Pi_m$ such that for each $h \in \{1, \ldots, m\}$:

- If $\mathsf{Q}_h = \mathsf{E}$, then every eventuality in Π_h has been marked in some node n_k for some k such that $j \leq k < i$.
- If $\mathsf{Q}_h = \mathsf{A}$ and $\mathsf{A}\Pi_h$ contains at least one eventuality, then $\mathsf{A}\Pi_h$ has been selected once in some node n_k such that $j \leq k < i$ and one of the eventualities in Π_h has been marked.

The procedure Eventuality_Selection performs in some fair way that ensures that any open branch will ever be *eventuality-covered*.

▶ **Definition 24** (Terminal Node)**.** A node n is terminal, if $\tau(n) = \Sigma_\perp$ or n is a loop-node of the branch b and b is eventuality-covered. Otherwise, we say that n is non-terminal.

Consequently, a non-terminal node is either a node that is not a loop-node or a loop-node whose branch is not eventuality-covered. A potentially-cycling formula could be selected more than once in a branch because the loop-node could change along the branch. In fact, the loop-node decreases in a well-founded order. The following example illustrates this issue.

▶ **Example 25.** Let $\sigma_1 = \mathsf{A}(\mathbf{T} \,\mathcal{U}\, (\neg c), \Box a)$, $\sigma_2 = \mathsf{A}(a \,\mathcal{U}\, \Box b)$, and $\sigma_3 = \mathsf{A}\Box c$, where $a, b, c \in \mathsf{Prop}$. Let $\Sigma_0 = \{\sigma_1, \sigma_2, \sigma_3\}$ be the initial set of state formulae. Let σ_1 be selected and $\mathbf{T} \,\mathcal{U}\, (\neg c)$ is the first marked eventuality. Then, $(\mathsf{A} \,\mathcal{U}\, \sigma)^+$ is applied to σ_1. Now in the node that

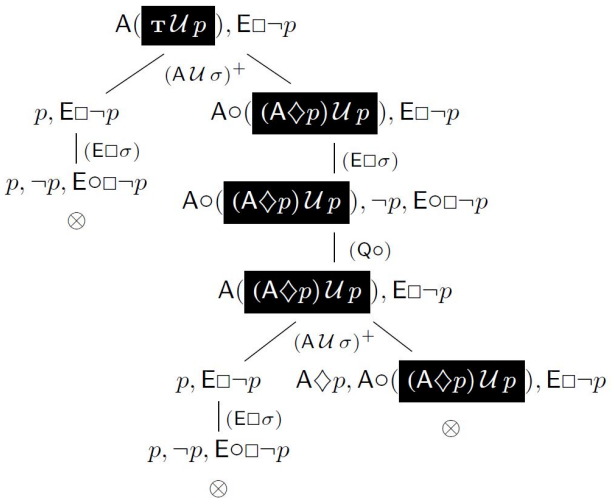

Figure 3 A closed tableau for $\mathsf{A}(\top \mathcal{U}\, p), \mathsf{E}\Box\neg p$. Marked eventualities are in black boxes, big circles represent AND-nodes or bunches. Whenever a bunch has a unique successor, we omit the big circle in the edge before the $(\mathsf{Q}\bigcirc)$-successor.

contains the next-step variant of σ_1, $(\mathsf{A}\Box\sigma)$ is applied to the next-step variant of σ_1; $(\mathsf{A}\,\mathcal{U}\,\Box)$ is applied to σ_2; and $(\mathsf{A}\Box\sigma)$ to σ_3. Then one of the open tableau branches is labelled by the elementary set: $\{a, c, \mathsf{A}(\bigcirc(((\neg\sigma_2)\vee(\neg\sigma_3)\vee a)\,\mathcal{U}\,(\neg c)), \bigcirc\Box a), \mathsf{A}\bigcirc(a\,\mathcal{U}\,\Box b), \mathsf{A}\bigcirc\Box c\}$. By rule $(\mathsf{Q}\bigcirc)$, we get a node n_1 labelled by the US $\tau(n_1) = \{\sigma_1', \sigma_2, \sigma_3\}$ where σ_1' is the selected formula: $\mathsf{A}(((\neg\sigma_2)\vee(\neg\sigma_3)\vee a)\,\mathcal{U}\,(\neg c), \Box a)$ and the eventuality $((\neg\sigma_2)\vee(\neg\sigma_3)\vee a)\,\mathcal{U}\,(\neg c)$ is kept marked. Hence, in one of the branches that enlarges n_1, the same sequence of rules produces a loop-node n_2 such that $\tau(n_2) = \tau(n_1)$. However, this branch is not eventuality-covered since σ_2 has not been selected yet. Then the selected formula in n_2 must be σ_2, and $(\mathsf{A}\,\mathcal{U}\,\Box)^+$ is applied to σ_2. Now, one of the open branches gets a node n_3 such that $\tau(n_3) = \{\sigma_1', \sigma_2', \sigma_3\}$ where $\sigma_2' = \mathsf{A}\Box b$. Then σ_1' is selected again. Note that the outer context of σ_1' has changed because σ_2 has been replaced by σ_2'. Hence, applying $(\mathsf{A}\,\mathcal{U}\,\sigma)^+$ to σ_1' and $(\mathsf{A}\Box\sigma)$ to σ_2' and σ_3, the leaf of one of the open branches is a node n_4 labelled by the US $\tau(n_4) = \{\sigma_1'', \sigma_2', \sigma_3\}$ where σ_1'' is the selected formula: $\mathsf{A}((((\neg\sigma_2)\wedge(\neg\sigma_2'))\vee(\neg\sigma_3)\vee a)\,\mathcal{U}\,(\neg c), \Box a)$. Note that $((\neg\sigma_2)\wedge(\neg\sigma_2'))\vee(\neg\sigma_3)\vee a$ is the conjunctive normal form of $((\neg\sigma_2)\vee(\neg\sigma_3)\vee a)\wedge((\neg\sigma_2')\vee(\neg\sigma_3)\vee a)$. Since, σ_1'' is kept selected, we will get a loop-node n_5 such that $\tau(n_5) = \tau(n_4)$ and now the branch is eventuality-covered. Indeed, this branch represents the following model of Σ_0: $\{a, c\}\langle\{a, b, c\}\rangle^\omega$.

▶ **Definition 26** (Bunch in a Tableaux, Closed Bunch and Tableaux)**.** A *bunch* b is a collection of branches that is maximal with respect to $(\mathsf{Q}\bigcirc)$-successor, i.e. every $(\mathsf{Q}\bigcirc)$-successor of any node in b are also in b. A bunch is closed if and only if at least one of its branches is closed. Otherwise it is open. A tableau is closed if and only if all its bunches are closed.

Any open tableau has at least one open bunch, formed by one or more open branches.

▶ **Example 27.** (Figure 3) In the applications of $(\mathsf{A}\,\mathcal{U}\,\sigma)^+$ rule, the inner context is empty; the outer context is $\mathsf{E}\Box\neg p$, its negation in nnf is $\mathsf{A}\Diamond p$. Hence, the label of the rightmost leaf, $\mathsf{A}\bigcirc((\mathsf{A}\Diamond p)\,\mathcal{U}\,p)$ is the simplification of the selected formula $\mathsf{A}\bigcirc(((\mathsf{A}\Diamond p)\wedge(\mathsf{A}\Diamond p))\,\mathcal{U}\,p)$.

▶ **Example 28.** On the left of Figure 4 we depict a representative open bunch of a tableau for the set of formulae: $p, \mathsf{A}\Box(\mathsf{E}\bigcirc p\wedge\mathsf{E}\bigcirc\neg p), \mathsf{A}(\Diamond\neg p, \Box p)$. We apply at once the Uniform_Tableau

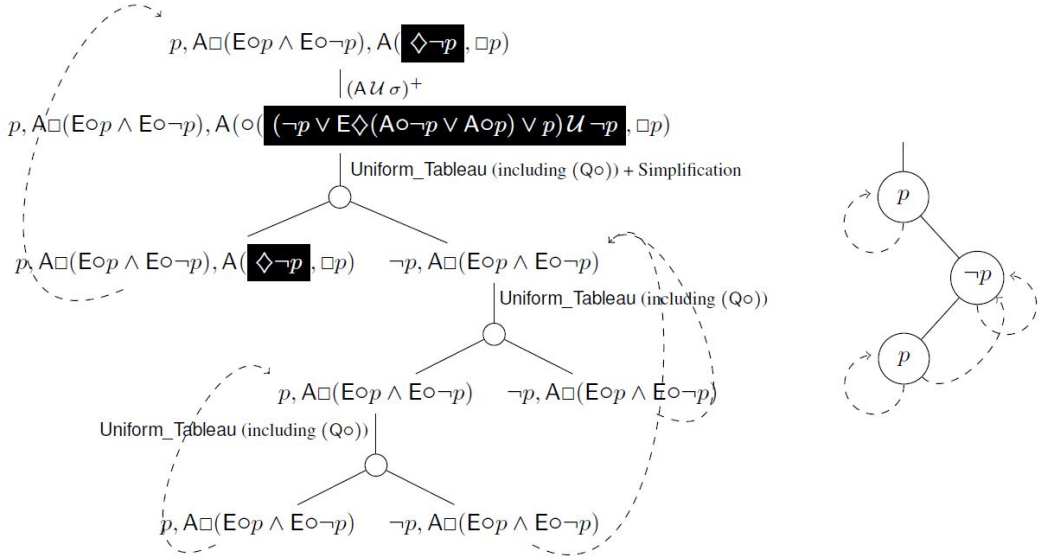

Figure 4 Open bunch in the tableau for $p, \mathsf{A}\Box(\mathsf{E}\bigcirc p \wedge \mathsf{E}\bigcirc\neg p), \mathsf{A}(\Diamond\neg p, \Box p)$ and represented model.

procedure subsequently choosing one of the leaves produced. For each node, we draw only one of the OR-children, but all the AND-children. In the marked eventuality, $\neg p \vee \mathsf{E}\Diamond(\mathsf{A}\bigcirc\neg p)$ comes from the negation of the outer context, and the disjunct p from the inner context. By "Simplification" $\neg p \vee \mathsf{E}\Diamond(\mathsf{A}\bigcirc\neg p \vee \mathsf{A}\bigcirc p) \vee p$ is reduced to $\mathbf{T}$ (in the left-hand child). In the right-hand node, $\neg p$ subsumes $\mathsf{A}((\ldots)\,\mathcal{U}\,\neg p, \Box p)$. This open bunch represents a model (of the input set of formulae) that we depict on the right of Figure 4.

5 Correctness: Soundness, Completeness and Termination

The soundness of our tableau method (Theorem 31) is proved on the basis that tableau rule preserve satisfiability (Lemma 30). For the latter it is essential to prove that the satisfaction of the negated inner context is preserved from segments of a limit path to the limit path itself (Proposition 29). The use of the formula φ_Π (Definition 17) is crucial for that.

▶ **Proposition 29** (Preservation of the Negated Inner Context). *Let Π be any set of basic path formulae and let φ_Π be as in Definition 17. Let $y = x_1^{\leq i_1} x_2^{\leq i_2} \cdots x_k^{\leq i_k} \cdots$ be a limit path in* $\mathsf{fullpaths}(\mathcal{K})$ *(of some Kripke structure $\mathcal{K}$). Then $\mathcal{K}, y \models \neg\pi$ holds for all $\pi \in \Pi$, provided that the following two conditions hold for all $n \geq 1$:*
(a) *$\mathcal{K}, x_1^{\leq i_1} x_2^{\leq i_2} \cdots x_n, j \models \neg\sigma_2$ for all $\sigma_1\,\mathcal{U}\,(\sigma_2 \wedge \Diamond\sigma_3) \in \Pi$ and all $j \in \{0..i_n\}$, and*
(b) *$\mathcal{K}, x_1^{\leq i_1} x_2^{\leq i_2} \cdots x_n^{\leq i_n}, i_n \models \neg\varphi_\Pi$.*

▶ **Lemma 30** (Soundness of the Tableau Rules). *For any set of state formulae Σ:*
(i) *For any α-formula α :* $\mathsf{Sat}(\Sigma, \alpha)$ *iff* $\mathsf{Sat}(\Sigma, S_\alpha)$.
(ii) *For any β-formula β of range k:* $\mathsf{Sat}(\Sigma, \beta)$ *iff* $\mathsf{Sat}(\Sigma, S_{\beta_i})$ *for some $1 \leq i \leq k$.*
(iii) *For any β^+-formula β of range k:* $\mathsf{Sat}(\Sigma, \beta)$ *iff* $\mathsf{Sat}(\Sigma, S^+_{\Sigma,\beta_i})$ *for some $1 \leq i \leq k$.*
(iv) *If Σ is a set of literals:* $\mathsf{Sat}(\Sigma, \mathsf{A}\bigcirc\Phi_1, \ldots, \mathsf{A}\bigcirc\Phi_n, \mathsf{E}\bigcirc\Psi_1, \ldots, \mathsf{E}\bigcirc\Psi_m)$ *iff for all $0 \leq i \leq m$:* $\mathsf{Sat}(\mathsf{A}\Phi_1, \ldots, \mathsf{A}\Phi_n, \mathsf{E}\Psi_i)$.

▶ **Theorem 31** (Soundness of the Tableau Method). *Given any set of state formulae Σ, if there exists a closed tableau for Σ then* $\mathsf{UnSat}(\Sigma)$.

In the rest of this section, we sketch the main stages of the proof of completeness of the presented tableaux method. Detailed proofs can be found in the technical report at `http://www.sc.ehu.es/jiwlucap/TechReport18.pdf` while relevant proofs of the most important Propositions, Lemmas and Theorems are given in the Appendix.

▶ **Definition 32** (Stage of a Tableaux). Given a branch, b, of a tableau T, a *stage* in T is every maximal subsequence of successive nodes $n_i, n_{i+1}, \ldots, n_j$ in b such that $\tau(n_k)$ is a not a (Q○)-child of $\tau(n_{k-1})$, for all k such that $i < k \leq j$. We denote by $\mathsf{stages}(b)$ the *sequence of all stages* of b.

▶ **Definition 33** ($\alpha\beta^+$-saturated Stage). A stage s in $\mathcal{A}^{sys}_{\Sigma}$ is $\alpha\beta^+$-*saturated* iff for all $\sigma \in \tau(s)$:

1. If σ is an α-formula then $S_\sigma \subseteq \tau(s)$
2. If σ is a β-formula of range k, but is not a β^+-formula, then $S_{\beta_i} \subseteq \tau(s)$ for some $1 \leq i \leq k$.
3. If σ is a β-formula and also a β^+-formula of range k then either $S_{\beta_i} \subseteq \tau(s)$ or $S^+_{\Sigma,\beta_i} \subseteq \tau(s)$ for some $1 \leq i \leq k$ and $\Sigma = \tau(n) \setminus \{\sigma\}$ for some $n \in s$.

▶ **Definition 34** (Expanded Bunch and Tableau). An open branch b is *expanded* if each stage $s \in \mathsf{stages}(b)$ is $\alpha\beta^+$-saturated and b is eventuality-covered. A bunch is expanded if all its open branches are expanded. A *tableau* is expanded if all its open bunches are expanded.

▶ **Proposition 35.** *(Trivial by construction) Given any set of state formulae Σ, the systematic tableau $\mathcal{A}^{sys}_{\Sigma}$ is expanded.*

▶ **Definition 36** (Open Bunch Model Construction). For any expanded bunch H of $\mathcal{A}^{sys}_{\Sigma}$, let $\mathcal{K}_H = (S, R, L)$ be a Kripke structure such that $S = \bigcup_{b \in H} \mathsf{stages}(b)$, R is the relation over $\mathsf{stages}(b)$ for any $b \in H$, and for any $s \in S$: $L(s) = \{p \mid p \in \tau(n) \cap \mathsf{Prop}$ for node $n \in s\}$.

▶ **Lemma 37** (Model Existence). *Let $\mathcal{A}^{sys}_{\Sigma}$ have an expanded bunch H and $\mathcal{K}_H = (S, R, L)$ be as in Definition 36. For every state $s \in S$, if $\sigma \in L(s)$ then $\mathcal{K}_H, s, 0 \models \sigma$. Therefore, for any expanded bunch H of $\mathcal{A}^{sys}_{\Sigma}$, $\mathcal{K}_H \models \Sigma$.*

▶ **Theorem 38** (Refutational Completeness). *Given any set of state formulae Σ, if $\mathsf{UnSat}(\Sigma)$ then there exists a closed tableau for Σ.*

▶ **Theorem 39** (Termination). *Given any set of state formulae Σ , the construction of the expanded tableau $\mathcal{A}^{sys}_{\Sigma}$ terminates.*

Finally Theorems 38 and 39 give us the desired completeness result stated in Theorem 40.

▶ **Theorem 40** (Completeness). *Given any set of state formulae Σ, if Σ is satisfiable then there exists a (finite) open expanded tableau for Σ.*

6 Conclusion

We introduced a new logic, ECTL$^{\#}$, in the family of BTL, which can represent a richer class of fairness constraints with the $\mathcal{U}$ operator. The tree-style one pass tableau method for ECTL$^{\#}$ handles inputs in an "analytic" way, due to the new, crucial for branching structures, concept of "inner context", in which eventualities are to be fulfilled. The tableau rules that invoke the inner context, are essential to handle A-disjunctive formulae. Our analysis of A-disjunctive and E-conjunctive formulae and of the prioritisation of eventualities, based on their structure and the context for their fulfillment, are important from the methodological point of view.

Our tableau technique is not directly extensible to CTL*. Without any significant modifications, β^+-rules become unsound for inputs that are beyond ECTL$^\#$ syntax due to nested path subformulae as in $\mathsf{A}\Diamond(\circ p \wedge \mathsf{E}\circ\neg p)$. To show the correctness of β^+-rules, we developed the technique to identify relevant state-formulae inside the specific path-modalities. This technique will be useful in studying more expressive logics (e.g. CTL*), as it allows to identify those subformulae that do not affect the "context", thus enabling the simplification of the structures.

The size of the systematic tableau for the input of size m is bounded by $2^{2^{O(m^2)}}$ (see technical report `http://www.sc.ehu.es/jiwlucap/TechReport18.pdf`). However, the method aims at the "shortest" way to fulfil the eventualities and, for many examples, finds the first open bunch, giving us a model for the tableau input. This significantly reduces the complexity. Finally, the presented technique is amenable for implementation – and this will be another stream of our future work. In the refinement and implementation of the algorithm we will be able to rely on similar techniques used in the implementation of its linear-time analogue.

References

1 Therese Berg and Harald Raffelt. Model checking. In Manfred Broy, Bengt Jonsson, Joost-Pieter Katoen, Martin Leucker, and Alexander Pretschner, editors, *Model-Based Testing of Reactive Systems*, pages 557–603. Springer-Verlag, Berlin Heidelberg, 2005. `doi:10.1007/b137241`.

2 Kai Brünnler and Martin Lange. Cut-free sequent systems for temporal logic. *Journal of Logic and Algebraic Programming*, 76(2):216–225, 2008. `doi:10.1016/j.jlap.2008.02.004`.

3 Edmund M. Clarke, E. Allen Emerson, and Aravinda P. Sistla. Automatic verification of finite-state concurrent systems using temporal logic specifications. *ACM Trans. Program. Lang. Syst.*, 8(2):244–263, 1986. `doi:10.1145/5397.5399`.

4 E. Allen Emerson. Temporal and modal logic. In Jan van Leeuwen, editor, *Handbook of Theoretical Computer Science (Vol. B)*, pages 995–1072. MIT Press, Cambridge, USA, 1990. URL: `http://dl.acm.org/citation.cfm?id=114891.114907`.

5 E. Allen Emerson and Joseph Y. Halpern. Decision procedures and expressiveness in the temporal logic of branching time. *Journal of Computer and System Sciences*, 30(1):1–24, 1985. `doi:10.1016/0022-0000(85)90001-7`.

6 E. Allen Emerson and Joseph Y. Halpern. Sometimes and not never revisited: On branching versus linear time temporal logic. *J. ACM*, 33(1):151–178, 1986. `doi:10.1145/4904.4999`.

7 E. Allen Emerson and Chin-Laung Lei. Temporal reasoning under generalized fairness constraints. In Monien B. and Vidal-Naquet G., editors, *STACS 1986, Lecture Notes in Computer Science, vol 210*, pages 21–37. Springer-Verlag Berlin Heidelberg, Karlsruhe, Federal Republic of Germany, 1986. `doi:10.1007/3-540-16078-7_62`.

8 Jose Gaintzarain, Montserrat Hermo, Paqui Lucio, Marisa Navarro, and Fernando Orejas. Dual systems of tableaux and sequents for PLTL. *Journal of Logic and Algebraic Programming*, 78(8):701–722, 2009. `doi:10.1016/j.jlap.2009.05.001`.

9 Rajeev Gore. Tableau methods for modal and temporal logics. In Marcello D'Agostino, Dov M. Dov Gabbay, Reiner Hähnle, and Joachim Posegga, editors, *Handbook of Tableau Methods*, pages 297–396. Springer, Netherlands, Dordrecht, 1999. `doi:10.1007/978-94-017-1754-0_6`.

10 Bernhard Josko. Model checking of ctl formulae under liveness assumptions. In Thomas Ottmann, editor, *Automata, Languages and Programming, 14th International Colloquium*,

pages 5–24. Springer-Verlag Berlin Heidelberg, Karlsruhe, Federal Republic of Germany, 1987. doi:10.1007/3-540-18088-5.

11 Jan Kretinsky and Ruslan Ledesma Garza. Rabinizer 2: Small deterministic automata for LTL\GU. In *Automated Technology for Verification and Analysis - 11th International Symposium, ATVA 2013*, pages 446–450, Heidelberg Dordrecht London New York, 2013. Springer. doi:10.1007/978-3-319-02444-8_32.

12 Nicolas Markey. Temporal logics. Course notes, Master Parisien de Recherche en Informatique, Paris, France, 2013. URL: http://www.lsv.ens-cachan.fr/Publis/PAPERS/PDF/NM-coursTL13.pdf.

13 Mark Reynolds. A tableau for CTL*. In Ana Cavalcanti and Dennis Dams, editors, *FM 2009: Formal Methods, Second World Congress, Eindhoven, The Netherlands, November 2-6, 2009. Proceedings*, volume 5850 of *Lecture Notes in Computer Science*, pages 403–418. Springer, 2009. doi:10.1007/978-3-642-05089-3_26.

14 Robert S. Streett and E. Allen Emerson. The propositional mu-calculus is elementary. In Jan Paredaens, editor, *Automata, Languages and Programming, 11th Colloquium, Antwerp, Belgium, July 16-20, 1984, Proceedings*, volume 172 of *Lecture Notes in Computer Science*, pages 465–472. Springer, 1984. doi:10.1007/3-540-13345-3_43.

15 Jun Sun, Yang Liu, Jin Song Dong, and Hai H. Wang. Specifying and verifying event-based fairness enhanced systems. In Shaoying Liu, Tom Maibaum, and Keijiro Araki, editors, *Formal Methods and Software Engineering: 10th International Conference on Formal Engineering Methods ICFEM 2008*, pages 5–24. Springer Science and Business Media, Kitakyushu-City, Japan, 2008. doi:10.1007/978-3-540-88194-0.

A Appendix

Proof of Proposition 14. Use Proposition 12 to construct a tableau with all its leaves labelled by elementary sets of formulae. Then apply the rule (Q○) to any leaf. Finally, apply (to every leaf) the rules (Eσ), (Aσ), ($\wedge$), and ($\vee$), as long as they are applicable. ◀

Proof of Proposition 29 (Preservation of the Negated Inner Context). We check the four cases of a basic path formula $\pi \in \Pi$. If π is of the form $\sigma_1 \mathcal{U} (\sigma_2 \wedge \Diamond\sigma_3)$, then property (a) ensures that every state in y satisfies $\neg\sigma_2$. Therefore, $\neg(\sigma_1 \mathcal{U} (\sigma_2 \wedge \Diamond\sigma_3))$ is satisfied in the limit path y. The remaining three cases are proved on the basis of (b) and Definition 17:
If $\pi = \Box(\sigma_1 \vee \Box\sigma_2)$, then $\mathcal{K}, x_1^{\leq i_1} x_2^{\leq i_2} \cdots x_n^{\leq i_n}, i_n \models \neg\sigma_1 \wedge \neg\sigma_2$ for all n. Therefore, it holds that $\mathcal{K}, y \models \neg\Box(\sigma_1 \vee \Box\sigma_2)$.
If $\pi = \Box(\sigma_1 \mathcal{U} \sigma_2)$, then $\mathcal{K}, x_1^{\leq i_1} x_2^{\leq i_2} \cdots x_n^{\leq i_n}, i_n \models \neg\mathsf{E}(\Diamond\sigma_2)$ for all n. Hence, $\mathcal{K}, y \models \neg\Box(\sigma_1 \mathcal{U} \sigma_2)$.
If $\pi = \sigma_1 \mathcal{U} \Box\sigma_2$, then $\mathcal{K}, x_1^{\leq i_1} x_2^{\leq i_2} \cdots x_n^{\leq i_n}, i_n \models \neg\sigma_2$ for all n. Hence, $\mathcal{K}, y \models \neg(\sigma_1 \mathcal{U} \Box\sigma_2)$. ◀

Proof of Lemma 30 (Soundness of Tableau Rules). Noting that (i), (i) and (iv) can be easily proved by the "systematic" application of the semantic definitions of temporal operators, we prove (iii). The "only if" direction" for each of the cases of β^+-rules is trivial.
For the "if" direction of rule $(\mathsf{E}\mathcal{U}\sigma)^+$, let us suppose that $\mathcal{K} \models \Sigma, \mathsf{E}(\sigma_1 \mathcal{U} (\sigma_2 \wedge \Diamond\sigma_3), \Pi)$. There exists $x \in \mathsf{fullpaths}(\mathcal{K})$ such that $\mathcal{K}, x, 0 \models \Sigma, \Pi, \sigma_1 \mathcal{U} (\sigma_2 \wedge \Diamond\sigma_3)$. We are going to prove that there exists $\mathcal{K}'$ such that one of the following two properties holds:
(a) $\mathcal{K}' \models \Sigma, \sigma_2, \mathsf{E}(\Diamond\sigma_3, \Pi)$
(b) $\mathcal{K}' \models \Sigma, \sigma_1, \mathsf{E}(\circ((\sigma \wedge \neg\Sigma) \mathcal{U} \sigma_2), \Pi)$.
If $\mathcal{K}, x, 0 \models \sigma_2 \wedge \Diamond\sigma_3$, then (a) is trivially satisfied for $\mathcal{K}' = \mathcal{K}$. Otherwise, for some $i > 0$, it holds that $\mathcal{K}, x, i \models \sigma_2 \wedge \Diamond\sigma_3$ and for all $j < i$: $\mathcal{K}, x, j \models \sigma_1$. Let j be the least number greater than 0 such that $\mathcal{K}, x, j \models \sigma_2 \wedge \Diamond\sigma_3$. Consider k to be the greatest index in $\{0, \ldots, j\}$ such that $\mathcal{K}, x, k \models \Sigma, \Pi$. If $k = j$, then (a) is satisfied for $\mathcal{K}' = \mathcal{K} \restriction x(k)$. Otherwise, if $k < j$

then $\mathcal{K}, x, k \models \Sigma, \sigma_1, \circ((\sigma_1 \wedge \neg\Sigma)\,\mathcal{U}\,(\sigma_2 \wedge \Diamond\sigma_3)), \Pi$. Hence, item (b) holds for $\mathcal{K}' = \mathcal{K} \restriction x(k)$. Rewriting in the above proof $\sigma_2 \wedge \Diamond\sigma_3$ by $\Box\sigma_2$, we obtain the proof for rule $(\mathsf{E}\,\mathcal{U}\,\Box)^+$. Indeed, both proofs for $(\mathsf{E}\,\mathcal{U}\,\sigma)^+$ and $(\mathsf{E}\,\mathcal{U}\,\Box)^+$ are very similar to the context-based rule in linear-time case of PLTL (see Lemma 5.1 in [8]).

For the "if" direction of rule $(\mathsf{A}\,\mathcal{U}\,\sigma)^+$, let us suppose that the three sets $\Sigma \cup S_{\Sigma,\beta_1}$, $\Sigma \cup S_{\Sigma,\beta_2}$, and $\Sigma \cup S_{\Sigma,\beta_3}$ of the rule $(\mathsf{A}\,\mathcal{U}\,\sigma)^+$ are unsatisfiable. We will show that the set $\Sigma, \mathsf{A}(\sigma_1\,\mathcal{U}\,(\sigma_2 \wedge \Diamond\sigma_3), \Pi)$ must be also unsatisfiable. By the hypothesis, we know that any model of Σ is not a model of S_{Σ,β_i} for all $i \in \{1,2,3\}$. In other words, for any $\mathcal{K}$ such that $\mathcal{K} \models \Sigma$, the followings three facts holds:

(a) $\mathcal{K} \not\models \sigma_2 \wedge \mathsf{A}(\Diamond\sigma_3, \Pi)$

(b) $\mathcal{K} \not\models \sigma_1 \wedge \mathsf{A}(\circ((\sigma_1 \wedge (\neg\Sigma \vee \varphi_\Pi))\,\mathcal{U}\,(\sigma_2 \wedge \Diamond\sigma_3)), \Pi)$

(c) $\mathcal{K} \not\models \mathsf{A}\Pi$

To show that $\Sigma, \mathsf{A}(\sigma_1\,\mathcal{U}\,(\sigma_2 \wedge \Diamond\sigma_3), \Pi)$ is unsatisfiable, we consider any $\mathcal{K}$ such that $\mathcal{K} \models \Sigma$ and prove that $\mathcal{K} \not\models \mathsf{A}(\sigma_1\,\mathcal{U}\,(\sigma_2 \wedge \Diamond\sigma_3), \Pi)$. Since $\mathcal{K} \models \Sigma$, then (a), (b) and (c) hold. According to (b), there are two possible cases:

(Case 1): If $\mathcal{K} \not\models \sigma_1$ then, by (a), either $\mathcal{K} \models \neg\sigma_1 \wedge \neg\sigma_2$ or $\mathcal{K} \models \neg\sigma_1 \wedge \mathsf{E}(\Box\neg\sigma_3, \neg\Pi)$. In both cases, it is easy to see that $\mathcal{K} \not\models \mathsf{A}(\sigma_1\,\mathcal{U}\,(\sigma_2 \wedge \Diamond\sigma_3), \Pi)$.

(Case 2): Otherwise, if $\mathcal{K} \not\models \mathsf{A}(\circ((\sigma_1 \wedge (\neg\Sigma \vee \varphi_\Pi))\,\mathcal{U}\,(\sigma_2 \wedge \Diamond\sigma_3)), \Pi)$, then we have that there exists $x_1 \in \mathsf{fullpaths}(\mathcal{K})$ such that $\mathcal{K}, x_1 \models \circ\neg((\sigma_1 \wedge (\neg\Sigma \vee \varphi_\Pi))\,\mathcal{U}\,(\sigma_2 \wedge \Diamond\sigma_3)) \wedge \neg\Pi$. This yields two possible cases:

(Case 2.1): If $\mathcal{K}, x_1 \models \circ\Box(\neg\sigma_2 \vee \Box\neg\sigma_3) \wedge \neg\Pi$, then it is trivial that $\mathcal{K} \not\models \mathsf{A}(\sigma_1\,\mathcal{U}\,(\sigma_2 \wedge \Diamond\sigma_3), \Pi)$.

(Case 2.2): Otherwise, there should exist $i_1 > 0$ that satisfies the following three properties:

(i) $\mathcal{K}, x_1, j \models (\neg\sigma_2) \vee \Box\neg\sigma_3$ for all j such that $0 \leq j \leq i_1$, and

(ii) $\mathcal{K}, x_1, i_1 \models \neg\sigma_1 \vee (\Sigma \wedge \neg\varphi_\Pi)$, and

(iii) $\mathcal{K}, x_1, 0 \models \neg\Pi$

If (i) is satisfied because $\mathcal{K}, x_1, j \models \Box\neg\sigma_3$ for some j such that $0 \leq j \leq i_1$, then trivially $\mathcal{K}, x_1, 0 \not\models \sigma_1\,\mathcal{U}\,(\sigma_2 \wedge \Diamond\sigma_3)$. This, along with the fact (iii), ensures that $\mathcal{K} \not\models \mathsf{A}(\sigma_1\,\mathcal{U}\,(\sigma_2 \wedge \Diamond\sigma_3), \Pi)$. Moreover, this also applies to any other formula $\sigma_1'\,\mathcal{U}\,(\sigma_2' \wedge \Diamond\sigma_3')$ in Π. Henceforth, in what follows, we can suppose that for all j such that $0 \leq j \leq i_1$: $\mathcal{K}, x_1, j \models \neg\sigma_2$ and also that $\mathcal{K}, x_1, j \models \neg\sigma_2'$ for all $\sigma_1'\,\mathcal{U}\,(\sigma_2' \wedge \Diamond\sigma_3') \in \Pi$.

If (ii) is satisfied because $\mathcal{K}, x_1, i_1 \models \neg\sigma_1$ then it is clear that $\mathcal{K}, x_1, 0 \not\models \sigma_1\,\mathcal{U}\,(\sigma_2 \wedge \Diamond\sigma_3)$. Therefore, by (i) and (iii), $\mathcal{K} \not\models \mathsf{A}(\sigma_1\,\mathcal{U}\,(\sigma_2 \wedge \Diamond\sigma_3), \Pi)$.

Otherwise, if (ii) is satisfied because $\mathcal{K}, x_1, i_1 \models \Sigma \wedge \neg\varphi_\Pi$, then, since $\mathcal{K}, x_1, i_1 \models \Sigma$, then again (a), (b) and (c) hold for $\mathcal{K} \restriction x_1(i_1)$ (instead of $\mathcal{K}$). Hence, reasoning for $\mathcal{K} \restriction x_1(i_1)$ as we do above for $\mathcal{K}$, there should exist a path $x_2 \in \mathsf{fullpaths}(\mathcal{K} \restriction x_1(i_1))$ such that one of the following two facts holds:

(Case 2.2.1): $\mathcal{K} \restriction x_1(i_1), x_2 \models \Box\neg(\sigma_2 \wedge \Diamond\sigma_3) \wedge \neg\Pi$, and therefore $\mathcal{K} \not\models \mathsf{A}(\sigma_1\,\mathcal{U}\,(\sigma_2 \wedge \Diamond\sigma_3), \Pi)$.

(Case 2.2.2): there should exist $i_2 > 0$ such that $\mathcal{K} \restriction x_1(i_1), x_2, i_2 \models \Sigma \wedge \neg\varphi_\Pi$ and for all $j \in \{0..i_2\}$:

- $\mathcal{K} \restriction x_1(i_1), x_2, j \models \neg\sigma_2$, and
- $\mathcal{K} \restriction x_1(i_1), x_2, j \models \neg\sigma_2'$ for all $\sigma_1'\,\mathcal{U}\,(\sigma_2' \wedge \Diamond\sigma_3') \in \Pi$j

Now, (a), (b) and (c) apply to $\mathcal{K} \restriction x_2(i_2)$. Hence, the infinite iteration of the second case yields a path $y = x_1^{\leq i_1} x_2^{\leq i_2} \cdots x_k^{\leq i_k} \cdots$ (that exists by the limit closure property) for which the Proposition 29 ensures that $\mathcal{K}, y \not\models \mathsf{A}(\sigma_1\,\mathcal{U}\,(\sigma_2 \wedge \Diamond\sigma_3), \Pi)$.

The proof for rule $(\mathsf{A}\,\mathcal{U}\,\Box)^+$ follows the same scheme. It is worth noting that, according to rule $(\mathsf{A}\,\mathcal{U}\,\Box)^+$, for all n: $\mathcal{K}, x_1^{\leq i_1} x_2^{\leq i_2} \cdots x_n^{\leq i_n}, i_{n+1} \models \neg\varphi_\Pi \wedge \neg\sigma_2$ because this ensures that the limit path y satisfies $\neg\Box\sigma_2$. ◀

Proof of Theorem 31 (Soundness of Tableau Method). Let T_Σ be a closed tableau for Σ. The set of formulas labelling at least one leaf in each bunch is inconsistent and therefore unsatisfiable. Then, by Lemma 30, the root Σ is unsatisfiable. ◀

To prove refutational completeness and termination, we first define a partial order relation on the set of basic state formulae. This order is the basis for inductively proving that $\mathcal{K}_H$ is a model of the tableau input (Lemma 37). The termination of Algorithm 1 (Theorem 39) is based on the extension that order to the set of finite sets of basic state formulae.

▶ **Definition 41** (Order on Basic state formulae)**.** The order $\preceq$ is defined as the reflexive-transitive closure of the smallest binary relation $\prec \subset \mathcal{F}_{\mathsf{Prop}} \times \mathcal{F}_{\mathsf{Prop}}$ that satisfies the following conditions:

1. $\mathsf{Q}\Pi' \prec \mathsf{Q}\Pi$ if $\Pi' \subset \Pi$.
2. $\sigma \prec \mathsf{Q}\Pi$ if σ is a proper state-subformula of $\mathsf{Q}\Pi$.
3. $\mathsf{Q}((\sigma_1 \wedge \delta)\,\mathcal{U}\,(\sigma_2 \wedge \Diamond\sigma_3), \Pi) \prec \mathsf{Q}(\sigma_1\,\mathcal{U}\,(\sigma_2 \wedge \Diamond\sigma_3), \Pi)$. In particular, $\mathsf{Q}((\sigma_1 \wedge \delta)\,\mathcal{U}\,\sigma_2, \Pi) \prec \mathsf{Q}(\sigma_1\,\mathcal{U}\,\sigma_2, \Pi)$.
4. $\mathsf{Q}(\Diamond\sigma_3, \Pi) \prec \mathsf{Q}(\sigma_1\,\mathcal{U}\,(\sigma_2 \wedge \Diamond\sigma_3), \Pi)$.
5. $\mathsf{Q}((\sigma_1 \wedge \delta)\,\mathcal{U}\,\Box\sigma_2, \Pi) \prec \mathsf{Q}(\sigma_1\,\mathcal{U}\,\Box\sigma_2, \Pi)$.
6. $\mathsf{Q}(\Box\sigma_2, \Pi) \prec \mathsf{Q}(\sigma_1\,\mathcal{U}\,\Box\sigma_2, \Pi)$.

where $\mathsf{Q} \in \{\mathsf{A}, \mathsf{E}\}$, Π, Π' are sets of basic path formulae, $\sigma, \sigma_1, \sigma_2$ are basic state formulae, and δ is a state formula different of the constant $\mathbf{T}$.
The extension $\prec^*$ is the partial order relation on finite sets of basic state formulae defined by $\Sigma_1 \preceq^* \Sigma_2$ if and only if for every $\sigma_1 \in \Sigma_1$ either $\sigma_1 \in \Sigma_2$ or there exists $\sigma_2 \in \Sigma_2$ such that $\sigma_1 \prec \sigma_2$. Therefore $\Sigma_1 \prec^* \Sigma_2$ if and only $\Sigma_1 \preceq^* \Sigma_2$ and $\Sigma_1 \neq \Sigma_2$.

Proof of Lemma 37 (Model Existence) (Sketch). Let $\mathcal{A}^{sys}_\Sigma$ have an expanded bunch H and $\mathcal{K}_H = (S, R, L)$ be as in Definition 36. We prove that, for every state $s \in S$, if $\sigma \in L(s)$ then $\mathcal{K}_H, s, 0 \models \sigma$. This proof is made by structural induction on the formula σ. That induction requires many auxiliary properties about how $\mathsf{ECTL}^{\#}$-formulae evolve along the branches in the systematic construction of the tableau $\mathcal{A}^{sys}_\Sigma$. These properties can be found in the technical report `http://www.sc.ehu.es/jiwlucap/TechReport18.pdf`). For space reasons, we only give here the following sketch of the proof. The base case, $\sigma = p \in \mathsf{Prop}$, is ensured by Definition 36. The bunch H ensures that, whenever a tableau node (at stage s) is labelled by an elementary set $\{\Sigma, \mathsf{A}\circ\Phi_1, \ldots, \mathsf{A}\circ\Phi_n, \mathsf{E}\circ\Psi_1, \ldots, \mathsf{E}\circ\Psi_m\} \subseteq L(s)$ then, by rule ($\mathsf{Q}\circ$), H contains one successor stage s_i (for each $i \in \{1, \ldots, m\}$) that, in turn, contains $\{A\Phi_1, \ldots, \mathsf{A}\Phi_n, \mathsf{E}\Psi_i\}$. By inductive hypothesis: $\mathcal{K}_H, s_i, 0 \models A\Phi_1, \ldots, \mathsf{A}\Phi_n, \mathsf{E}\Psi_i$, for each $i \in \{1, \ldots, m\}$. Therefore, $\mathcal{K}_H, s, 0 \models \{\Sigma, \mathsf{A}\circ\Phi_1, \ldots, \mathsf{A}\circ\Phi_n, \mathsf{E}\circ\Psi_1, \ldots, \mathsf{E}\circ\Psi_m\}$. Every branch $b \in H$ is open, so it is a cyclic branch such that $\mathsf{path}(b) = s_0, s_1, \ldots, s_{i-1}\langle s_i, s_{i+1}, \ldots, s_j\rangle^\omega$. For any $s_k \in \mathsf{path}(b)$ and any formula $\mathsf{Q}\Pi$ in $\tau(s_k)$, we prove that $\mathcal{K}_H, s_k, 0 \models \mathsf{Q}\Pi$, by induction in $\mathsf{Q}\Pi$. Formulae of the highest priority have to be selected at the stage previous to s_i. Hence, the first node of stage s_i is a loop-node that must be labelled by a set exclusively formed by potentially-cycling and cycling formulas (in particular, the empty set) ◀

Proof of Theorem 38 (Refutational Completeness). Suppose that for an input Σ there is no closed tableau. Then the systematic tableau $\mathcal{A}^{sys}_\Sigma$ would be open and there would be at least one expanded bunch H in $\mathcal{A}^{sys}_\Sigma$. By Lemma 37, $\mathcal{K}_H \models \Sigma$. Consequently, Σ would be satisfiable. ◀

Simplification rules are very useful to reduce the tableau construction but, more importantly, some simplification rules are essential for termination.

▶ **Definition 42** (Simplification Rules). First, to stop the growth of the subformula σ in the successive next-step variants $(\sigma_1 \wedge \sigma)\,\mathcal{U}\,\varphi$, we use trivial simplification rules such as $\varphi \wedge \varphi \longrightarrow \varphi$ and $\varphi \vee \varphi \longrightarrow \varphi$, as well as classical subsumptions rules. Second, to simplify the detection of equal node labels (for looping in tableau branches) we use the following rules:
$(\sqsubset \mathsf{E}\Box\,\mathcal{U}\,)$ $\mathsf{E}(\sigma_1\,\mathcal{U}\,\sigma_2, \Box(\sigma_1\,\mathcal{U}\,\sigma_2), \Pi) \longrightarrow \mathsf{E}(\Box(\sigma_1\,\mathcal{U}\,\sigma_2), \Pi)$.
$(\sqsubset \mathsf{A}\Box\,\mathcal{U}\,)$ If $\Pi' \subseteq \Pi$ then $\mathsf{A}(\sigma_1\,\mathcal{U}\,\sigma_2, \Pi) \wedge \mathsf{A}(\Box(\sigma_1\,\mathcal{U}\,\sigma_2), \Pi') \longrightarrow \mathsf{A}(\Box(\sigma_1\,\mathcal{U}\,\sigma_2), \Pi')$.
Finally, to prevent the duplications of the original eventuality $\sigma_1\,\mathcal{U}\,\sigma_2$ and its successive next-step variants by rules $(\mathsf{Q}\Box\,\mathcal{U}\,)$ and $(\mathsf{Q}\,\mathcal{U}\,\sigma)^+$, and to ensure termination, we use the following rules:
$(\sqsubset \mathsf{A}\sigma\,\mathcal{U}\,)$ $\sigma_2 \wedge \mathsf{A}(\sigma_1\,\mathcal{U}\,\sigma_2, \Pi) \longrightarrow \sigma_2$.
$(\sqsubset \mathsf{E}\,\mathcal{U}\,\sigma)$ $\mathsf{E}((\sigma_1 \wedge \sigma)\,\mathcal{U}\,\varphi, \sigma_1\,\mathcal{U}\,\varphi, \Pi) \longrightarrow \mathsf{E}((\sigma_1 \wedge \sigma)\,\mathcal{U}\,\varphi, \Pi)$
$(\sqsubset \mathsf{A}\,\mathcal{U}\,\sigma)$ If $\Pi' \subseteq \Pi$ then $\mathsf{A}((\sigma_1 \wedge \sigma)\,\mathcal{U}\,\varphi, \Pi') \wedge \mathsf{A}(\sigma_1\,\mathcal{U}\,\varphi, \Pi) \longrightarrow \mathsf{A}((\sigma_1 \wedge \sigma)\,\mathcal{U}\,\varphi, \Pi')$.
In order to prove termination, the following Propositions 43 and 44 ensure that any open branch of $\mathcal{A}^{sys}_{\Sigma}$ cannot be labelled by an infinitely $\preceq^*$-decreasing succession of set of formulae in the sense of items 3 and 5 in Definition 41. In addition, Proposition 45 shows that any open branch of $\mathcal{A}^{sys}_{\Sigma}$ is eventuality-covered.

▶ **Proposition 43.** *Let b be an open branch of $\mathcal{A}^{sys}_{\Sigma}$, let $s_i \in \mathsf{stages}(b)$ and let $\Sigma \cup \{\mathsf{E}\Pi\}$ be the US labelling the first node of s_i where $\mathsf{E}\Pi$ has been selected and one of the eventualities $\pi_{\mathcal{U}} \in \Pi$ has been marked. Then there exists some $k \geq i$ such that:*
(a) *If $\pi_{\mathcal{U}} = \sigma_1\,\mathcal{U}\,(\sigma_2 \wedge \Diamond\sigma_3)$ then $\{\sigma_2, \mathsf{E}(\Diamond\sigma_3, \Pi')\} \subseteq \tau(s_k)$ for some Π' such that $\mathsf{E}(\Diamond\sigma_3, \Pi') \prec \mathsf{E}\Pi$.*
(b) *If $\pi_{\mathcal{U}} = \sigma_1\,\mathcal{U}\,\Box\sigma_2$ then $\mathsf{E}(\Box\sigma_2, \Pi') \in \tau(s_k)$ for some Π' such that $\mathsf{E}(\Box\sigma_2, \Pi') \prec \mathsf{E}\Pi$.*
(c) *If $\pi_{\mathcal{U}} = \Box(\sigma_1\,\mathcal{U}\,\sigma_2)$ then $\sigma_2 \in \tau(s_k)$.*

Proof. Suppose that the rule $(\mathsf{E}\,\mathcal{U}\,\sigma)^+$ (or $(\mathsf{E}\,\mathcal{U}\,\Box)^+$) has been successively applied, keeping the selection, between stages s_i and s_k. In the case (c), $(\mathsf{E}\,\mathcal{U}\,\sigma)^+$ is applied immediately after $(\mathsf{E}\Box\,\mathcal{U}\,)$. Then, in any of the three cases, the US labelling the first node of stage s_k has the form $\Sigma_{s_k}, \mathsf{E}((\sigma_1 \wedge (\neg\Sigma_{s_i} \wedge \cdots \wedge \neg\Sigma_{s_{k-1}}))\,\mathcal{U}\,\varphi, \Pi')$ where φ is either $(\sigma_2 \wedge \Diamond\sigma_3)$ or $\Box\sigma_2$ or σ_2; each Σ_{s_j} $(i \leq j \leq k)$ is the context of the selected formula at the first node of each stage s_j (in particular $\Sigma_{s_i} = \Sigma$); and Π' is such that : $\mathsf{E}((\sigma_1 \wedge (\neg\Sigma_{s_i} \wedge \cdots \wedge \neg\Sigma_{s_{k-1}}))\,\mathcal{U}\,\varphi, \Pi') \prec \mathsf{E}\Pi$. Since no other β^+-rule is applied between stage s_i and s_k, each Σ_{s_j} is a subset of the finite set formed by all state formulae that are subformulae of some formula in $\Sigma_{s_i} \cup (\Pi \setminus \{\pi\})$ and their negations.[8] Hence, there is a finite number of different Σ_{s_j}. Therefore, after a finite number applications of the β^+-rule, the label containing the set $\{\sigma_1 \wedge (\neg\Sigma_{s_i} \wedge \cdots \wedge \neg\Sigma_{s_{k-1}}), \Sigma_{s_k}\}$ must be inconsistent. Henceforth, the open branch b must satisfy (a) or (b) or (c), depending on the case of $\pi_{\mathcal{U}}$, and according to the rules $(\mathsf{E}\,\mathcal{U}\,\sigma)^+$ and $(\mathsf{E}\,\mathcal{U}\,\Box)^+$. ◀

For A-disjunctive formulae, not only the outer context, but also the inner context plays an important role. The proof of the next proposition is based on the use of both kinds of context.

▶ **Proposition 44.** *Let b be an open branch of $\mathcal{A}^{sys}_{\Sigma}$, let $s_i \in \mathsf{stages}(b)$ and let $\Sigma \cup \{\mathsf{A}\Pi\}$ be the US labelling the first node of s_i where $\mathsf{Q}\Pi$ has been selected and one of the eventualities $\pi_{\mathcal{U}} \in \Pi$ has been marked. Then there exists some stage $s_k \in \mathsf{stages}(b)$ (for some $k \geq i$) such that one of the following two facts holds:*

[8] This finite set can be seen as a "local closure".

(a) *if* $\pi_{\mathcal{U}} = \sigma_1 \mathcal{U} (\sigma_2 \wedge \Diamond\sigma_3)$ *then* $\{\sigma_2, \mathsf{A}(\Diamond\sigma_3, \Pi')\} \in \tau(s_k)$ *for some* Π' *such that* $\mathsf{E}(\Diamond\sigma_3, \Pi') \prec \mathsf{E}\Pi$*; if* $\pi = \sigma_1 \mathcal{U} \Box\sigma_2$*, then* $\mathsf{A}(\sigma_2, \Pi') \in \tau(s_k)$ *for some* Π' *such that* $\mathsf{A}(\Box\sigma_2, \Pi') \prec \mathsf{A}\Pi$*; and if* $\pi_{\mathcal{U}} = \Box(\sigma_1 \mathcal{U} \sigma_2)$ *then* $\sigma_2 \in \tau(s_k)$*, or*

(b) *the first node* n *of stage* s_k *is a loop-node that contains* $\{\mathsf{A}((\sigma_1 \wedge \delta) \mathcal{U} \varphi, \Pi')$ *for some* δ*, some* φ *that depends on* $\pi_{\mathcal{U}}$*, and some* Π' *such that* $\{\mathsf{A}((\sigma_1 \wedge \delta) \mathcal{U} \varphi, \Pi') \prec \mathsf{A}\Pi$*.*

Proof. Suppose that the rule $(\mathsf{A}\mathcal{U}\sigma)^+$ (or $(\mathsf{A}\mathcal{U}\Box)^+$) has been successively applied, keeping the selection, between stages s_i and s_k. Then the US labelling the first node of stage s_k has the form:

$$\Sigma_{s_k}, \mathsf{A}((\sigma_1 \wedge (\neg\Sigma_{s_i} \vee \varphi_{\Pi_{s_i}}) \wedge \cdots \wedge (\neg\Sigma_{s_{k-1}} \vee \varphi_{\Pi_{s_{k-1}}})) \mathcal{U} \varphi, \Pi_{s_k})$$

where φ is $\sigma_2 \wedge \Diamond\sigma_3$ or $\Box\sigma_2$ or σ_2 (depending on the case of $\pi_{\mathcal{U}}$); and each Σ_{s_j} and each Π_{s_j} $(i \leq j \leq k)$ are respectively the outer context and the inner context at the first node of each stage s_j. In particular, $\Sigma_{s_i} = \Sigma$ and $\Pi_{s_i} = \Pi$. Since no other β^+-rule is applied between s_i and s_k, then[9]

- each Σ_{s_j} is a subset of the finite set formed by all state formulae that are subformulae of some formula in $\Sigma \cup (\Pi \setminus \{\pi_{\mathcal{U}}\})$, their negations, and a formula $\mathsf{E}(\mathbf{T}\mathcal{U}\sigma_2)$ for each subformula $\Box(\sigma_1 \mathcal{U} \sigma_2)$ in $\Sigma \cup (\Pi \setminus \{\pi_{\mathcal{U}}\})$ (see Definition 17), and
- each Π_{s_j} is a subset of the finite set of all path formulae that are subformulae of some formula in Π.

Therefore, since we simplify $\varphi \wedge \varphi$ by φ and $\varphi \vee \varphi$ by φ, after a finite number β^+-rule applications (if b is not closed) some previous node label should be repeated. ◀

▶ **Proposition 45.** *Any open branch* b *of* $\mathcal{A}^{sys}_{\Sigma}$ *is eventuality-covered.*

Proof Sketch. This sketch is based on several properties of the systematic tableau construction which, for space reasons, can only be found in the technical report `http://www.sc.ehu.es/jiwlucap/TechReport18.pdf`. Whenever we say *by construction* we refer to some of these properties. Let b be any open branch of $\mathcal{A}^{sys}_{\Sigma}$. By construction, there exists some stage $s_i \in \mathsf{stages}(b)$ such that the label of the first node n in s_i is either empty or a non-empty US Σ that consists exclusively of potentially-cycling and cycling formulae (and possibly a set of literals). If Σ is empty or a set of literals, b is finished by a cycle consisting of two empty labels, which is trivially eventuality-covered. Otherwise, Σ contains at most one formula $\mathsf{E}\Pi$ along with a set $\mathsf{A}\Pi_1, \ldots, \mathsf{A}\Pi_k, \mathsf{A}\Pi'_1, \ldots, \mathsf{A}\Pi'_m$ such that for all $1 \leq i \leq k$: $\mathsf{A}\Pi_i$ is a potentially-cycling formula, and for all $1 \leq i \leq m$: $\mathsf{A}\Pi'_i$ is a cycling formula. Note that if there is one $\mathsf{E}\Pi$ in Σ, then $\mathsf{E}\Pi$ is a cycling formula. If $k = 0$ and $\Pi'_1, \ldots, \Pi'_m, \Pi$ does not contain any eventuality (as formula or subformula). By construction, the first-node of the next-stage s_{i+1} is a loop-node also labelled by Σ and b is eventuality-covered. Otherwise, one of the formulae in Σ (the label of node n) is selected, namely σ, and one of its eventualities, namely π, is marked. By construction, the node n should be followed (in b) by some node n', labelled by Σ', such that either π is fulfilled in Σ' or n' is a loop-node. In both cases $\Sigma' \prec^* \Sigma$. If Σ' does not contain more eventualities (apart from π), b is already eventuality-covered. Otherwise, the selection of eventualities is applied in the first state of the new stage. All the formulae of highest-priority are "solved" firstly (if Σ' have some), until the label is again a set of potentially-cycling and cycling formulae. Then we change to select a formula (if any) that has been not already selected in the branch (if it is not a loop-node) or in the loop (if it is a loop-node). This process gives –as node labels– sequences of sets of

[9] Finite sets Σ_{s_j} and Π_{s_j} can be seen as "local closures".

cycling and potentially-cycling formulae that are strictly decreasing with respect to $\preceq^*$. Henceforth, while b remains open, loop-nodes strictly decrease w.r.t. $\preceq^*$. Since the number k of potentially-cycling formulae is finite, after a finite number of stages, we get a minimal loop-node labelled by some Σ''. If Σ'' does not contain any eventuality, the Uniform_Tableau produces a new node labelled by Σ''. Otherwise, any selection made on this Σ'' leads to a stage whose first node is also labelled by Σ''. Hence, after selecting all the selectable formulae in Σ'' the branch b is eventuality-covered. ◀

Proof of Theorem 39 (Termination). Tableau rules produce a finite branching, hence König's Lemma applies. The subsumption-based simplification rules (Definition 42) do prevent the generation of formulae containing the original eventuality when a next-step variant of the eventuality has been generated. By Propositions 43 and 44, once one eventuality is marked, a kind of "local closure" allows us to ensure the finiteness of the application of a β^+-rule to the selected eventuality. Finally, Proposition 45 ensures that any open branch is eventually-covered. ◀