Sublogics of a Branching Time Logic of Robustness

John M^{c}Cabe-Dansted[a], Clare Dixon[b,*], Tim French[a], Mark Reynolds[a]

[a] *Computer Science and Software Engineering (M002), The University of Western Australia, 35 Stirling Highway, Crawley WA 6009 Perth, Australia*
[b] *Department of Computer Science, University of Liverpool, Ashton Street, Liverpool, L69 3BX, United Kingdom*

Abstract

In this paper we study sublogics of RoCTL*, a recently proposed logic for specifying robustness. RoCTL* allows specifying robustness in terms of properties that are robust to a certain number of failures. RoCTL* is an extension of the branching time logic CTL* which in turn extends CTL by removing the requirement that temporal operators be paired with path quantifiers. In this paper we consider three sublogics of RoCTL*. We present a tableau for RoBCTL*, a bundled variant of RoCTL* that allows fairness constraints to be placed on allowable paths. We then examine two CTL-like restrictions of CTL*. Pair-RoCTL* requires a temporal operator to be paired with a path quantifier; we show that Pair-RoCTL* is as hard to reason about as the full CTL*. State-RoCTL* is restricted to State formulas, and we show that there is a linear truth preserving translation of State-RoCTL into CTL, allowing State-RoCTL to be reasoned about as efficiently as CTL.

Keywords: RoCTL*, Bundles, Tableau, CTL
2010 MSC: 03B70, 68Q17

1. Introduction

RoCTL* [1] is temporal logic for specifying *Robustness*. As with other simpler propositional temporal logics such as the popular linear LTL [2], it can be used to reason about the behaviour of finite state transition systems, and hardware or software systems that can be modelled with these. Temporal connectives, such as $\bigcirc$ for next step (which indicates that its argument is true at the next step), and $\mathcal{U}$ for until (a bimodal operator that indicates that one argument is true until the other is satisfied), added to the classical propositional ones, allow us to specify and deduce properties of such systems as they operate

*Corresponding author
Email addresses: john.mccabe-dansted@uwa.edu.au (John M^{c}Cabe-Dansted), CLDixon@liverpool.ac.uk (Clare Dixon), tim.french@uwa.edu.au (Tim French), mark.reynolds@uwa.edu.au (Mark Reynolds)

over time. The branching logics CTL [3] and CTL* [4] add the ability to reason about alternative possible steps. RoCTL*, goes even further, and adds basic deontic operators to consider what the system should do, versus what it may do if some undesirable transitions occur. These transitions may represent occurrences such as packet loss which although unwanted must be accounted for in the design of the system. The robustness operators, allow us to consider overall behaviour of the system if only a limited number of undesired transitions occur.

Thus, RoCTL* has three path quantifiers: All Paths (**A**), Obligatory (**O**), and Robustly (▲). From these operators the duals ExistPath (**E**), Permissible (**P**) and Prone (△) are defined. This extends the full Computation Tree Logic (CTL*) whose only path quantifier is **A** (and its dual **E**).

As an example, consider the following specification in RoCTL* which uses the usual LTL temporal connective $\Box$ for always. The formula $\mathbf{O}\blacktriangle\Box(p \rightarrow \Diamond q)$ says that it is obligatory that robustly always a p request is eventually followed by a q grant. This means that along every failure-free behaviour, even if there is one failure deviating from such a behaviour then any p event is eventually followed some time in the future by a q event. In earlier work [1, 5] we have showed that this language can comfortably and effectively express situations with conflicting resource requirements, temporary communication failures and contrary-to-duty obligations (what a system should do when it hasn't done what it should do). More examples can be found in Section 3.

The logic RoCTL* was introduced in [1] but more detailed technical investigation of the language and its computational properties was reported in [6]. That paper showed that the satisfiability problem for RoCTL* is decidable. It provided truth and satisfiability preserving translations of RoCTL* into Quantified Computation Tree Logic (QCTL*) [7, 8] and Full Computation Tree Logic (CTL*), either of which can be used to decide the satisfiability of RoCTL* formulas. However, it was also shown that translating RoCTL* formulas into tree automata results in at least a singly exponentially blowup per alternation with the ▲ operator [6]. Thus, there is no elementary reduction of RoCTL* into tree automata or CTL*. Nor is there any known elementary decision procedure.

In Section 4 below we will argue that RoCTL* is not significantly harder than CTL* to decide, in practice. However, for many purposes, decision procedures even for CTL* are too computationally demanding and, since RoCTL* is a conservative extension of CTL*, it is clear that RoCTL* is at least as hard to decide as CTL*.

We clearly have a problem with advocating direct use of the RoCTL* language for applications. This leads us to make the common move of considering restricted sub-languages of RoCTL* and considering whether some sufficiently expressive sub-languages may have easier decision problems. Thus the current paper considers sub-logics of RoCTL* from the point of view of being amenable to automated reasoning as well as their expressiveness. For example, we will discuss how restricting the syntax of RoCTL* to state-formulas makes the model checking and satisfiability problems as easy as those for CTL.

We will investigate semantic and syntactic sublogics of RoCTL*. RoBCTL* is a so-called *bundled* variant of RoCTL* : *limit closure* is not valid in RoBCTL*,

while it is valid in RoCTL*, making RoBCTL* a semantic sublogic. Informally the limit closure property states that if we have an infinite sequence of states σ such that for all n the first n states form an allowed path, then the infinite sequence σ forms an allowed path. This property ensures that all paths through the structure are allowed in CTL*, while in BCTL* we are limited to some bundle of paths which may or may not include all paths through the structure. We will also investigate two CTL-like restrictions of RoCTL*.

We explain the concept of a bundled variant of a branching time logic in the following section. Bundled branching-time logics adopt a different semantics to the non-bundled versions and have been studied for example in [9]. The basic idea is that in the bundled variant, not all paths through the structure must be considered when evaluating a formula, just those in the relevant bundle which is a set of allowed paths. RoCTL* can be thought of as the restriction of RoBCTL* where the bundle must include all possible paths in the structure. Under this interpretation, RoBCTL* and RoCTL* are not just trivially expressively equivalent but truly equivalent when interpreted over RoCTL* structures. The bundled variant has a set of valid formulas which is a subset of those allowed in of RoCTL*. Similarly, any formula that is satisfiable in RoCTL* is satisfiable in RoBCTL*. We investigate RoBCTL* for two reasons.

The first reason is because it is generally easier to find tableaux for bundled logics; for example, note that the tableau for BCTL* [9] was found before the tableaux for CTL* [10, 11]. The bundled variant is of interest as BCTL* has a simple tableau with optimal (2EXPTIME) worst case performance, and a tree based implementation that performs much better in practice than one would expect from the worst case performance bounds [12]. By comparison the only decision procedure for CTL* that combines good real world performance with optimal worst case results is neither a simple tableau or automata technique, but rather a combination of a tableau and a parity game solver [11, 13].

The second reason is bundled logics can express fairness constraints which are commonly needed for specifications [14, 15]. In a robust system there is often an implicit fairness constraint. For example, a normally functioning network will lose a packet with some probability; however, (almost surely) given enough retransmits the packet will eventually get through. The direct specification of this in CTL* or RoCTL* contradicts the limit-closure property. This is because the limit closure property means that if it is always possible that we lose a packet, it is also possible that we always lose a packet.

By contrast, while RoBCTL* allows models that disallow paths that do not satisfy fairness constraints. Note that (Ro)BCTL* are not designed to represent traditional numerical probabilities, for examples bundles cannot distinguish between $p = 0.1$ and $p = 0.99$.

In Section 4 we will present a tableau based decision procedure for the bundled variant RoBCTL* of RoCTL*, and show that under certain reasonable restrictions on the nesting of operators, the tableau has elementary worst-case complexity.

The other sub-languages we investigate are syntactical restrictions of RoCTL* using the original semantics. Even if an elementary decision procedure is found

for RoCTL*, it is clear that it will be at least as complex as CTL*. Despite the expressive power of CTL*, the less expressive CTL is frequently used; the most famous example of the use of CTL is the 1995 verification of the Futurebus+ cache coherence protocol [16]. However, later papers have also tended to also use approaches based on CTL rather than the full CTL* [17, 18]. This is because the satisfiability and model-checking decision problems for CTL are much easier. The CTL* decision problems are exponentially harder [19] than the CTL decision problems [20], the satisfiability of CTL* formulas is 2-EXPTIME complete while testing the satisfiability of CTL formulas is EXPTIME-COMPLETE; model checking for CTL* is singly exponential in the length of the formula, but for CTL we can model check in time linear in the length of the formula; in either case we can model check in time linear in the size of the model [21, 22, 23] (that is, the sum of the number of nodes and edges).

We will examine two CTL-like restrictions of RoCTL*. Arguably the most intuitive CTL-like restriction of RoCTL* is to require (like in CTL) that each of the LTL operators ($\bigcirc$ and $\mathcal{U}$) be paired with a path quantifier ($\mathbf{A}$, $\mathbf{O}$, or $\blacktriangle$). This restriction is called Pair-RoCTL. However, it will be shown in Section 5 that this restriction of RoCTL* has the same expressivity as RoCTL* and it will similarly be shown that Pair-RoCTL is as at least as hard to decide as CTL*. The difficulty in reasoning with Pair-RoCTL comes from the fact that $\blacktriangle\phi$ is not a state formula. That is the truth of $\blacktriangle\phi$ may depend on which future eventuates, not just the current state. Thus another restriction of RoCTL* called State-RoCTL will be examined. This restriction instead pairs the LTL operators with a non-empty sequence of path operators which must form a state formula. Hence $\mathbf{O}\triangle\blacktriangle(\phi\mathcal{U}\psi)$ is a State-RoCTL formula but $\blacktriangle(\phi\mathcal{U}\psi)$ is not. It will be shown in Section 6 that we can use standard CTL decision procedures to reason about State-RoCTL, although we need to inspect the internal state of the CTL model checking algorithm of [24] to achieve the same order of complexity as the original. Thus although Pair-RoCTL is an intuitive definition of a CTL-like restriction of RoCTL*, State-RoCTL is more CTL-like in complexity than Pair-RoCTL.

Every property that can be expressed in State-RoCTL can be expressed in CTL, yet State-RoCTL can naturally express interesting robustness properties. CTL is significantly less expressive than CTL* and RoCTL*. For example, CTL cannot express fairness, the property that if a process is ready to run infinitely often, then it will be chosen to run infinitely often [4]. Nevertheless State-RoCTL is expressive enough to capture some interesting properties of RoCTL*, such as direct alternations between $\blacktriangle$ and $\triangle$. Additionally, the truth preserving translation of State-RoCTL into CTL results in a formula that may be exponentially longer than the original. A linear translation will be given that is both satisfiability preserving and computationally efficient, providing efficient decision procedures for State-RoCTL, but this translation adds atoms and is not truth preserving.

With current technology State-RoCTL is more tractable for larger formulas than CTL*; we have given simple translations into CTL and there are a number of easily available and fast decision procedures for CTL. The resolution

procedure CTL-RP [25] can solve the decision problems relating to the (coordinated attack) problem presented in [1] in under 2 seconds. The Tableau Work Bench [26] also has little difficulty with translations of our examples into CTL, determining that the specification for feeding a cat in Example 22 is satisfiable almost instantaneously. We also note that dedicated CTL solvers tend to be more efficient [27] than the CTL* prover proposed in [11, 13].

1.1. Our contributions

The main contributions of this paper are: (Section 4) a tableau for RoBCTL*, which terminates but can require a non-elementary amount of time; (Section 5) a proof that the satisfiability and model checking problems for Pair-RoCTL are at least as hard as those for CTL*; and (Section 6) a polynomial reduction of State-RoCTL to CTL. This reduction shows that the model checking problems for State-RoCTL have the same complexity as CTL, and so out of the sublogics of RoCTL* considered in this paper, State-RoCTL is the easiest to reason about. This is the first appearance of these results in a journal.

The core results in this paper have been presented in abridged form at conferences [28, 29]. The tableau for RoBCTL* originally appeared in [28]; in this paper we additionally include a proof of soundness and completeness of this tableau. Partial proofs relating to Pair-RoCTL appeared in [29], but only considered the $\mathcal{U}$ and $\mathbf{E}$ operators. Here we included full proofs and also consider the expressivity of Pair-RoCTL. State-RoCTL also appeared in [29], here we provide more details on the proofs and consider corollaries such as model checking. Section 7 on related work was not published in the conferences.

We use a slightly different definition of RoCTL* structures in this paper than some previous papers. RoCTL* structures distinguish between successful and failing transitions. This can be done naturally by having two accessibility relations, one for failure transitions, and one for success transitions [1]. However, when discussing translations to CTL and CTL* it is helpful to restrict ourselves to traditional CTL/Kripke models that only have one accessibility relation. For any RoCTL* structure we can (1) split worlds which have both incoming success and failure transitions, (2) put a special "violation" atom at the end of failing transitions, and merge the success and failure accessibility relation. We call the resulting logic over Kripke structure $\text{RoCTL}^*_{\mathbf{v}}$.

In this paper we define RoCTL* as the restriction of $\text{RoCTL}^*_{\mathbf{v}}$ that prohibits the violation atom occurring in formulas. This change in notation does not affect which formulas are satisfiable. Given the translation between the models is trivial and linear, this difference does not affect other properties such as model checking complexity either, so we consider this an alternate notation for the same logic. For a more thorough discussion of RoCTL* and related logics, see the PhD thesis [5]. For a more thorough discussion of related work, see Section 7.

2. RoBCTL*, RoCTL*, CTL* and CTL

In this section we define RoBCTL*, RoCTL*, CTL* and CTL. We first provide some basic intuitions and definitions starting with our set of variables.

Deontic logics allow us to express what ought to be. The simple modal logic known as "KD" or "K" divides worlds/states into allowable states where all obligations are fulfilled, and disallowed states that fail some obligation. It may be more natural to talk about allowed and disallowed actions.

The intuition behind RoCTL* is that there are two types of transitions: failure transitions and success transitions. A failure transition is a possible transition where some form of failure occurs, for example losing a packet. A success transition represents a transition that occurs when the system is acting as it should. This was how RoCTL* was defined in [1]. In this paper we compare RoCTL* to CTL*. To do this we need to define RoCTL* on CTL* structures. As CTL* only has a single transition relation, we will instead use a special atom $\mathbf{v}$, used to indicate that the last transition was a failure. To convert an original RoCTL* structure to this new CTL* based RoCTL* structure, we can unroll the structure into a tree so that every state has at most one possible parent, and add the $\mathbf{v}$ atom to the states reached by failure transitions. For a full proof of equivalence of these definitions see [5].

The RoCTL* logic is designed to represent robust systems. That is, systems that ensure that some goals are met even in the face of a certain number of failures occurring. Note that a failure transition may not involve the system failing to achieve any of its goals. For example, even a real time system may be able to recover fully from some fixed number of packets losses, and a single bit error in ECC memory may cause no harm. A particular failure transition may happen to have a beneficial effect, such as flipping a bit that has already been flipped once. However, given an assumption that the number of failures is no more than some number n, we may be able to verify that a system achieves some/all of its goals.

Definition 1. We let $\mathbb{V}$ be our set of variables. The set $\mathbb{V}$ contains a special variable $\mathbf{v}$. A valuation g is a map from a set of worlds S to the power set of the variables.

The statement $p \in g(w)$ is to be read as "the variable p is true at world w". The $\mathbf{v}$ atom will be used to represent failure transitions. In the terminology of Deontic logic, it is forbidden to follow a path with a failure transition. While the interpretation of forbidden depends on the domain of application, from the point of view of robust systems it is natural to think of a failure transition being the result of an event not desired by the designer of the system, but that the designer wishes to account for, such as a corrupted network packet or mistake by a user of the system.

Informally it may be possible to enter a state labelled with $\mathbf{v}$, but it is forbidden to do so; entering such a state will be considered a failure.

As is normal we say a binary relation is serial if every element has a successor.

Definition 2. We say that a binary relation R on S is serial (total) if for every a in S there exists b in S such that aRb.

While in some logics the truth of formulas depends solely on the current world, the truth of CTL* and BCTL* (and hence RoCTL* and RoBCTL*) may depend on which future eventuates. These futures are represented as infinitely long (full) paths through the structure. For this reason, we provide a formal definition of fullpaths.

Definition 3. We call an ω-sequence $\sigma = \langle w_0, w_1, \ldots \rangle$ of worlds a fullpath iff for all non-negative integers i we have $w_i R w_{i+1}$. For all i in $\mathbb{N}$ we define $\sigma_{\geq i}$ to be the fullpath $\langle w_i, w_{i+1}, \ldots \rangle$, we define σ_i to be w_i and we define $\sigma_{\leq i}$ to be the sequence $\langle w_0, w_1, \ldots, w_i \rangle$.

Note that the indices in the definition above start at 0 and can be thought of as a distance. We now define allowable sets of fullpaths called bundles.

Definition 4. We say that a set of fullpaths Π is suffix closed iff for all $\pi \in \Pi$ and positive integers i we have $\pi_{\geq i} \in \Pi$. We say that a set of fullpaths Π is fusion closed iff for any pair of non-negative integers i, j and fullpaths $\sigma, \pi \in \Pi$ such that $\sigma_i = \pi_j$ we have $\sigma_{\leq i} \cdot \pi_{\geq j+1} \in \Pi$. We say a non-empty set of fullpaths B through (S, R) is a bundle of (S, R) iff B is suffix and fusion closed.

Motivation for bundled logics has been given in other publications. For example, it can be difficult to adapt some satisfiability checking techniques to non-bundled logics with the limit-closure property [20], and certain fairness constraints can be naturally expressed in a restricted bundled logic BCTL which has singly exponential satisfiability checking procedure [14, 15]. We will explain the need for suffix and fusion closure in terms of the semantics and satisfiable formulas in Example 12.

The difference between "bundled" and non-"bundled" logics is that unbundled logics do not need to be limit closed. The limit closure property is rarely used explicitly, as bundled logics do not require that their bundles be limited closed. Unbundled logics can be defined more simply without using bundles but rather just quantifying over all paths; however, the semantics of CTL* can be equivalently defined using bundles that are also required to be limit closed. To illustrate the difference between bundled and non-bundled logics we will now define limit closure. Informally, limit closure means that if every prefix of a path is in the bundle, the path is in the bundle.

Definition 5. We say that a set of fullpaths Π is limit closed when for every fullpath σ, if for all $i \in \mathbb{N}$ there exists $\pi \in \Pi$ such that $\pi_{\leq i} = \sigma_{\leq i}$ then $\sigma \in \Pi$.

We now provide a definition of a structure.

Definition 6. A BCTL-structure $M = (S, R, g, B)$ is a 4-tuple containing a set of worlds S, a serial binary relation R on S, a valuation g on the set of worlds S and a bundle B on (S, R). We say M is a RoBCTL-structure if every world has an allowed successor, that is for all $x \in S$ there exists $(x, y) \in R$ such that $\mathbf{v} \notin g(y)$.

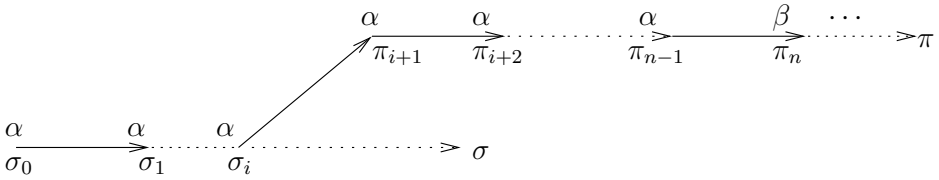

Figure 1: Example of Deviation π satisfying $\alpha \mathcal{U} \beta$

We now define the property of failure-freeness. This means that, in the future, no failing transitions are taken. Informally, a failure-free fullpath represents a perfect future. This is somewhat similar to a very simple traditional deontic logic [30, 31] called Standard Deontic Logic (SDL). SDL is the modal logic K with the addition of seriality on the Krikpe semantics, meaning that what is morally necessary (or obligatory) is also permissible. Whereas the Obligatory operator in Standard Deontic Logic quantifies over acceptable worlds, the Obligatory operator we will define quantifies over failure-free fullpaths.

Definition 7. We say that a fullpath σ is failure-free iff for all $i > 0$ we have $\mathbf{v} \notin g(\sigma_i)$. We define $\delta^\omega(w)$ to be the set of all fullpaths in B starting with world w and $\delta^0(w)$ to be the set of all failure-free fullpaths in B starting with w. We call a BCTL-structure a RoBCTL-structure iff $\delta^0(w)$ is non-empty for every $w \in S$.

We choose the notation above as δ^ω includes paths with potentially an infinite number of failures and δ^0 only includes paths without any failures.

We will now define deviations. We will use the notation δ^+ as informally deviations represent the possibility of adding an additional failure to some step i along a path. After i we follow a different path, and we allow only a single failure not on the existing path so no failures occur after $i + 1$.

Deviations are intended to represent possible failures we may wish to be able to recover from, and if our system is robust to failures we also want it to be robust in the face of correct transitions. For this reason we allow the new transition added at step i to be a success as well as a failure. In addition the robustly operator we will define quantifies over the present path as well as deviations from that path. This is because it would be odd to say that a goal is robustly achieved if it will not be achieved. For example, in Figure 3 all deviations (and deviations of deviations etc.) of σ reach p, but not σ itself. It would be odd to say that σ robustly ensures that we would eventually reach a p. These conventions are also followed by previous work on RoCTL* [1].

Definition 8. For two fullpaths σ and π in the bundle B we say that π is an i-deviation from σ iff $\sigma_{\leq i} = \pi_{\leq i}$ and $\pi_{\geq i+1} \in \delta^0(\pi_{i+1})$. We say that π is a deviation from σ if there exists a non-negative integer i such that π is an i-deviation from σ. We define a function δ^+ from fullpaths to sets of fullpaths such that where σ and π are fullpaths in B, the fullpath π is in $\delta^+(\sigma)$ iff π is a deviation from σ.

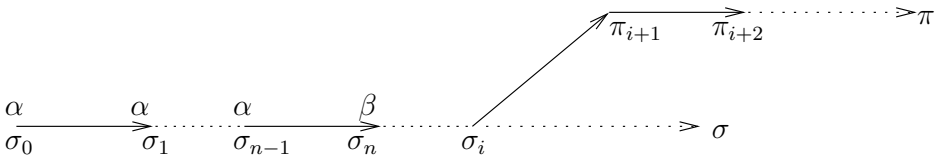

Figure 2: Path σ satisfying $\alpha \mathcal{U} \beta$, and π which deviates after β occurs

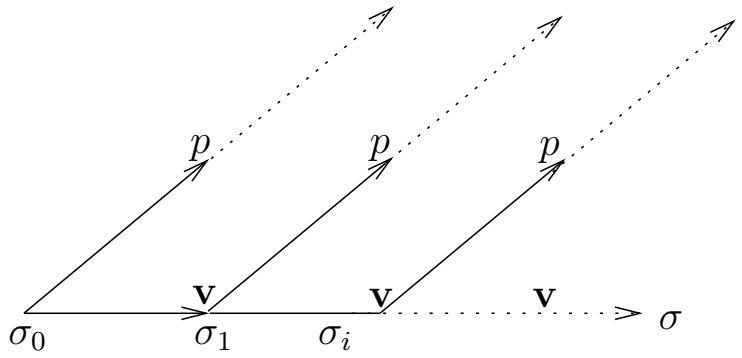

Figure 3: Example where p occurs on all deviations of σ, but not σ itself.

Note that deviations always have a finite number of failures along the path, since i is always finite.

For an example of a formula being satisfied along a deviation, see Figure 1. In this example $\alpha \mathcal{U} \beta$ is satisfied along the deviation as α is satisfied along the original path up to σ_i, where the deviation occurs, and then $\alpha \mathcal{U} \beta$ continues to be satisfied along the deviation. More trivial examples are also possible. In Figure 2 we see that we reach β before the deviation occurs, so $\alpha \mathcal{U} \beta$ is likewise satisfied along the original path.

We see that $\delta^0(\sigma_0) \subseteq \delta^+(\sigma) \subseteq \delta^\omega(\sigma_0)$. Bundled logics are semantic variants that share the original syntax. Where p varies over $\mathbb{V}$, we define the well formed formulas of RoCTL* and RoBCTL* formulas according to the following abstract syntax:

$$\phi := \alpha \mid \neg\phi \mid (\phi \wedge \phi) \mid (\phi \mathcal{U} \phi) \mid \bigcirc\phi \mid \blacktriangle\phi$$
$$\alpha := p \mid \neg\alpha \mid (\alpha \wedge \alpha) \mid \mathbf{A}\phi \mid \mathbf{O}\phi \ .$$

The α formulas above are called state formulas. Formulas that are not state formulas are called path formulas. To allow for alternative semantic interpretations of RoBCTL* and RoCTL* (such as in [32]), we do not consider a formula that explicitly contains $\mathbf{v}$ to be a RoBCTL* formula, although our results work equally well for such formulas and the behaviour of the $\mathbf{O}$ and $\blacktriangle$ operators do depend on $\mathbf{v}$. We call the logic without the restriction on $\mathbf{v}$ appearing in formulas RoBCTL$^*_{\mathbf{v}}$.

The $\neg$, $\wedge$, $\bigcirc$, $\mathcal{U}$ and $\mathbf{A}$ are the familiar "not", "and", "next", "until" and "all paths" operators from CTL.

Definition 9. We say that a pair of formulas ϕ, ψ are equivalent ($\phi \equiv \psi$) iff for all structures M and paths σ through M:

$$M, \sigma \vDash \phi \iff M, \sigma \vDash \psi \ .$$

We now define the abbreviations in terms of the base operators as follows: $\bot \equiv (p \wedge \neg p)$, $\top \equiv \neg\bot$, $\phi \vee \psi \equiv \neg(\neg\phi \wedge \neg\psi)$, $\Diamond\phi \equiv (\top \mathcal{U} \phi)$, $\Box\phi \equiv \neg\Diamond\neg\phi$, $\phi \mathcal{W} \psi \equiv (\phi \mathcal{U} \psi) \vee \Box\phi$, $\mathbf{E}\phi = \neg\mathbf{A}\neg\phi$, $\phi \rightarrow \psi \equiv (\neg\phi \vee \psi)$ and $\phi \leftrightarrow \psi \equiv (\phi \rightarrow \psi) \wedge (\psi \rightarrow \phi)$ are defined as in CTL*. As with Standard Deontic Logic (SDL, also known as KD or D) [30] logic, we define $\mathbf{P}\phi \equiv \neg\mathbf{O}\neg\phi$. Note that unlike $\mathbf{A}\phi$ and $\mathbf{E}\phi$, a formula of the form $\blacktriangle\phi$ is a path formula rather than a state-formula. Finally, we define the dual $\triangle$ of $\blacktriangle$ as the abbreviation $\triangle\phi \equiv \neg\blacktriangle\neg\phi$. We call the $\mathbf{O}$, $\mathbf{P}$, $\blacktriangle$, $\triangle$ operators Obligatory, Permissible, Robustly and Prone respectively.

In this paper we will also consider two syntactic sublogics of RoCTL*, "Pair-RoCTL" and "State-RoCTL". These will be discussed in Sections 5 and 6.

Definition 10. The syntax of Pair-RoCTL is defined by the following:

$$\begin{aligned} \phi &:= \phi \wedge \phi \mid \neg\phi \mid p \mid \mathbf{A}\psi \mid \mathbf{O}\psi \mid \blacktriangle\psi \\ \psi &:= \phi \overline{\mathcal{U}} \phi \mid \phi \mathcal{U} \phi \mid \bigcirc\phi \end{aligned}$$

The Release operator $\overline{\mathcal{U}}$ above is defined such that $\phi \overline{\mathcal{U}} \psi \equiv \neg(\neg\phi \mathcal{U} \neg\psi)$. Note that the addition of $\overline{\mathcal{U}}$ to the BNF above is a convenience so that we do not have to include the three path operators $\mathbf{E}$, $\mathbf{P}$, and $\triangle$ as $\mathbf{E}(\phi \mathcal{U} \psi)$, $\mathbf{P}(\phi \mathcal{U} \psi)$ and $\triangle(\phi \mathcal{U} \psi)$ can be represented as $\neg\mathbf{A}(\neg\phi \overline{\mathcal{U}} \neg\psi)$, $\neg\mathbf{O}(\neg\phi \overline{\mathcal{U}} \neg\psi)$ and $\neg\blacktriangle(\neg\phi \overline{\mathcal{U}} \neg\psi)$ respectively.

Definition 11. We define State-RoCTL formulas as follows:

$$\begin{aligned} \alpha &:= \mathbf{A}\theta \mid \mathbf{E}\theta \mid \mathbf{E}\bigcirc\alpha \mid \mathbf{O}\theta \mid \mathbf{P}\theta \mid \mathbf{P}\bigcirc\alpha \mid \alpha \wedge \alpha \mid \neg\alpha \mid p \\ \theta &:= \triangle\theta \mid \blacktriangle\theta \mid \alpha \mathcal{U} \alpha \end{aligned}$$

Formulas of the form θ will not be called State-RoCTL formulas, instead they will be called State-RoCTL path-formulas. In State-RoCTL the distinction between state and path formulas is important. We will use the symbols α and β to refer to state formulas in this section. The symbol θ will be used to refer to a path-formula, and the symbols ϕ and ψ will be used to refer to formulas that may be either state or path formulas. Additionally where we could write something like $M, \sigma_{\geq i} \vDash \alpha$ we will instead write $M, \sigma_i \vDash \alpha$ to remind ourselves that, as α is a state formula, the choice of the remainder of the path is irrelevant to the truth of α.

We define similar abbreviations to CTL, for example we use $\mathbf{O}\bigcirc\alpha$ as an abbreviation for $\neg\mathbf{P}\bigcirc\neg\alpha$. Additionally, note that the first transition of a deviation can be any success or failure transition leading away from the current

node. As such it is clear that $\triangle\bigcirc\alpha \leftrightarrow \mathbf{E}\bigcirc\alpha$ and $\blacktriangle\bigcirc\alpha \leftrightarrow \mathbf{A}\bigcirc\alpha$ are valid for any state formula α. Thus we treat $\triangle\bigcirc\alpha$ and $\blacktriangle\bigcirc\alpha$ as abbreviations for $\mathbf{E}\bigcirc\alpha$ and $\mathbf{A}\bigcirc\alpha$ respectively. Note that these last two abbreviations are not valid for RoCTL*, $\triangle\bigcirc\alpha$ is only equivalent to $\mathbf{E}\bigcirc\alpha$ because in State-RoCTL we know that α is a state-formula.

We define truth of a RoBCTL* formula ϕ on a fullpath $\sigma = \langle w_0, w_1, \ldots\rangle$ in a RoBCTL-structure M recursively as follows:

$$\begin{aligned}
M,\sigma &\vDash \bigcirc\phi \text{ iff } M,\sigma_{\geq 1} \vDash \phi\\
M,\sigma &\vDash \phi\mathcal{U}\psi \text{ iff } \exists i \in \mathbb{N}, \text{ s.t. } M,\sigma_{\geq i} \vDash \psi \text{ and;}\\
&\qquad\qquad \forall j \in \mathbb{N},\, j < i \implies M,\sigma_{\geq j} \vDash \phi\\
M,\sigma &\vDash \mathbf{A}\phi \text{ iff } \forall \pi \in \delta^{\omega}(\sigma_0),\, M,\pi \vDash \phi\\
M,\sigma &\vDash \mathbf{O}\phi \text{ iff } \forall \pi \in \delta^{0}(\sigma_0),\, M,\pi \vDash \phi\\
M,\sigma &\vDash \blacktriangle\phi \text{ iff } \forall \pi \in \delta^{+}(\sigma),\, M,\pi \vDash \phi \text{ and } M,\sigma \vDash \phi\ .
\end{aligned}$$

The definitions for p, $\neg$ and $\wedge$ are as we would expect from classical logic.

$$\begin{aligned}
M,\sigma &\vDash p \text{ iff } p \in g(\sigma_0)\\
M,\sigma &\vDash \neg\phi \text{ iff } M,\sigma \nvDash \phi\\
M,\sigma &\vDash \phi \wedge \psi \text{ iff } M,\sigma \vDash \phi \text{ and } M,\sigma \vDash \psi\ .
\end{aligned}$$

The intuition behind the $\blacktriangle$ operator is that it quantifies over paths that could result if a single failure was introduced; the deviations have at most one failure not on the original path, and they are identical to the original path until this failure occurs. The definition of bundles used in this paper have been used previously in the definition of BCTL* (see for example [9]); however, we will now show that some intuitively unsatisfiable formulas would be satisfiable if we did not require bundles to be suffix and fusion closed.

Example 12. If we relaxed the suffix closure requirement, then the formula $\mathbf{E}\bigcirc(\bigcirc p \wedge \mathbf{A}\bigcirc\neg p)$ would be satisfiable. If not for suffix closure we could have a path $\sigma \in B$ such that $M,\sigma \vDash \bigcirc\bigcirc p$ but not have any path $\sigma_{\geq 1} \in B$ starting at σ_1 such that $\sigma_{\geq 1} \vDash \bigcirc p$. If the fusion closure requirement was relaxed, the formula $\mathbf{A}\bigcirc\bigcirc p \wedge \mathbf{E}\bigcirc\mathbf{E}\bigcirc\neg p$ would be satisfiable. Using fusion closure we see that if there are paths $\sigma, \pi \in B$ such that $\pi_0 = \sigma_1$ and $\pi_0 \vDash \bigcirc\neg p$ then there must exist a path $\sigma_0 \cdot \pi$ which starts at σ_0 and $M,(\sigma_0 \cdot \pi) \vDash \bigcirc\bigcirc p$, whereas without fusion closure there would be no requirement that such a path exists. Note that since we do not require that the bundles be limit closed, the limit closure axiom [33] of CTL* is not valid in BCTL*:

$$\mathbf{A}\Box(\mathbf{E}\alpha \rightarrow \mathbf{E}\bigcirc(\mathbf{E}\beta\mathcal{U}\mathbf{E}\alpha)) \quad\rightarrow\quad (\mathbf{E}\alpha \rightarrow \mathbf{E}\Box(\mathbf{E}\beta\mathcal{U}\mathbf{E}\alpha))\ .$$

Definition 13. A function τ from formulas to formulas is truth-preserving iff for all M,σ and ϕ it is the case that $M,\sigma \vDash \phi \iff M,\sigma \vDash \tau(\phi)$. We say that a function τ from formulas to formulas is satisfiability-preserving iff: $\exists M,\sigma$ s.t. $M,\sigma \vDash \phi \iff \exists M',\sigma'$ s.t. $M',\sigma' \vDash \tau(\phi)$.

Given that traditional modal logics define truth at worlds, instead of over paths, many important properties of modal logics assume such a definition of truth. When dealing with those properties we can use the following definition of truth of RoBCTL* formulas at worlds.

Definition 14. A RoBCTL* state formula is true at a world if it is true on some path leading from that world, or more formally:

$$M, w \vDash \alpha \text{ iff } \exists \pi \text{ s.t. } \pi_0 = w \text{ and } M, \pi \vDash \alpha \ .$$

For the purposes of this paper it is most natural to define RoCTL* as RoBCTL* with an additional restriction of the models.

Definition 15. A RoCTL-structure (S, R, g, B) is a RoBCTL-structure where B contains every possible path through R. Likewise, a CTL-structure is a BCTL-structure where B contains every possible path through R.

Note that the original definition of RoCTL-structures [6] did not have B, but was rather a 3-tuple (S, R, g). Since B is determined by R in a RoCTL-structure, we can consider (S, R, g) as an abbreviation of (S, R, g, B). This leaves the definition of RoCTL* in this paper equivalent to the previous one.

We define CTL* to be the syntactic restriction of RoCTL* without **O** or ▲.

Definition 16. Where p varies over $\mathbb{V}$, we define CTL* formulas according to the following abstract syntax:

$$\phi := p \,|\, \neg\phi \,|\, (\phi \wedge \phi) \,|\, (\phi \mathcal{U} \phi) \,|\, \bigcirc\phi \,|\, \mathbf{A}\phi \ .$$

CTL* was proposed as an extension to CTL, which in turn was proposed as an extension of UB Logic that added an until operator [20]. However, here it is more convenient to define CTL as a syntactic restriction of CTL*, with the following syntax [3]:

$$\phi ::= p \,|\, \neg\phi \,|\, (\phi \wedge \phi) \,|\, \mathbf{A}\,(\phi \mathcal{U} \phi) \,|\, \mathbf{E}\bigcirc\phi \,|\, \mathbf{E}\,(\phi \mathcal{U} \phi) \ .$$

In CTL, we treat $\mathbf{A}\bigcirc\phi$, $\mathbf{A}\Diamond\phi$ and $\mathbf{A}\,(\phi \mathcal{W} \psi)$ as abbreviations of $\neg\mathbf{E}\bigcirc\neg\phi$, $\mathbf{A}\,(\top \mathcal{U} \phi)$ and $\neg\mathbf{E}\,((\neg\psi)\,\mathcal{U}\,(\neg\phi \wedge \neg\psi))$ respectively. Likewise, in CTL, we treat $\mathbf{E}\Diamond\phi$ and $\mathbf{E}\,(\phi \mathcal{W} \psi)$ as abbreviations of $\mathbf{E}\,(\top \mathcal{U} \phi)$ and $\neg\mathbf{A}\,(\neg\psi \mathcal{U}\,(\neg\phi \wedge \neg\psi))$. Finally, we define $\mathbf{A}\Box\phi$ and $\mathbf{E}\Box\phi$ as abbreviations of $\neg\mathbf{E}\Diamond\neg\phi$ and $\neg\mathbf{A}\Diamond\neg\phi$.

The following lemma will not be used in any proofs. It is included to make the subtle distinction between bundled versus unbundled logics more clear and in particular when they are interchangeable.

Lemma 17. *Given a formula ϕ that does not contain $\mathcal{U}$ (or the abbreviations $\Diamond$ and $\Box$ that use $\mathcal{U}$), ϕ will be satisfiable in RoCTL* iff it is satisfiable in RoBCTL*.*

Proof. It is clear that limit closure only has an effect on infinitely long paths. For a more detailed proof, see the Section 3.3.4 of the PhD thesis [5]. □

We now define some operators for comparing formulas.

Definition 18. For any pair of formulas (ϕ, ψ), we say that $\phi \sqsubseteq \psi$ iff ϕ is a subformula of ψ. As normal where S is a set we define the size $|S|$ of S as the number of elements of S. For a formula ϕ, we define the length $|\phi|$ of ϕ to be the total number of occurrences of symbols in the representation of ϕ excluding parentheses. For example, $|(p \wedge p)|$ is three as p occurs twice and $\wedge$ occurs once.

Proposition 19. RoCTL* (RoBCTL*) is a conservative extension of CTL* (BCTL*).

If a BCTL* formula ϕ is satisfiable in BCTL* then there is a BCTL-structure M that satisfies ϕ and does not have the special atom $\mathbf{v}$. As M does not contain the special atom $\mathbf{v}$, we see that the BCTL-structure is a RoBCTL-structure.

If a BCTL* formula ϕ is satisfiable in RoBCTL*, then there exists a RoBCTL-structure M that satisfies ϕ. Since RoBCTL-structures are a subclass of BCTL-structures, M is also a BCTL-structure. We see that M satisfies ϕ under the BCTL* semantics as they are the same as the RoBCTL* semantics for the BCTL* operators. The same argument can be made for RoCTL*/RoCTL*.

Theorem 20. *When interpreted over Ro(B)CTL-structures, BCTL$^*_{\mathbf{v}}$ is expressively equivalent to RoBCTL$^*_{\mathbf{v}}$.*

Proof. First it is clear that as BCTL$^*_{\mathbf{v}}$ is just a syntactic restriction of RoBCTL$^*_{\mathbf{v}}$, it cannot have greater expressive power. It is trivial to see that

$$\mathbf{O}\phi \quad \equiv \quad \mathbf{A}\left(\bigcirc\Box\neg\mathbf{v} \rightarrow \phi\right).$$

With a bit more effort it can be shown that the ▲ operator can also be translated into BCTL$^*_{\mathbf{v}}$ [6]. This provides a truth preserving translation from RoBCTL* into BCTL*. □

The same argument can also be used to show that RoCTL$^*_{\mathbf{v}}$ has the same expressive power as CTL$^*_{\mathbf{v}}$.

Proposition 21. When interpreted over Ro(B)CTL-structures, RoCTL* (RoBCTL*) has the same effective expressive power as CTL* (BCTL*).

Recall that RoCTL* is defined here as a syntactic restriction of RoCTL$^*_{\mathbf{v}}$ that is not allowed to include the special atom $\mathbf{v}$ in formulas. So clearly RoCTL$^*_{\mathbf{v}}$ is at least as expressive as RoCTL* (technically it is strictly more expressive as $\mathbf{v}$ cannot be translated into RoCTL*). Thus CTL$^*_{\mathbf{v}}$ is also more expressive than CTL*. Since RoCTL* is a syntactic extension of CTL* it is intuitive that it would be at least as expressive as CTL*. Indeed interpreted over a CTL-structure without any appearance of the special atom $\mathbf{v}$ in the valuation of any states, RoCTL* become equivalent to CTL*, and the $\mathbf{O}$ and ▲ operators become equivalent to $\mathbf{A}$. Thus from the above arguments and that RoCTL$^*_{\mathbf{v}}$ has the same expressive power as CTL$^*_{\mathbf{v}}$ we see that the expressive power of RoCTL* is between (or the same as) that of CTL$^*_{\mathbf{v}}$ and CTL* (without $\mathbf{v}$).

However, since $\text{CTL}^*_{\mathbf{v}}$ does not have any special operators that rely on the special atom $\mathbf{v}$, we see that $\mathbf{v}$ is only special in that it is guaranteed to satisfy $\mathbf{A}\Box\mathbf{E}\bigcirc\neg\mathbf{v}$. Satisfying $\mathbf{A}\Box\mathbf{E}\bigcirc\neg\mathbf{v}$ isn't very exciting, particularly since it is satisfied by any structure that does not have $\mathbf{v}$ in the valuation of any states. Essentially $\text{CTL}^*_{\mathbf{v}}$ is strictly more expressive than CTL* (without $\mathbf{v}$) since it uses one more atom than CTL* however there is no interesting difference in their expressive powers. Thus RoCTL*, CTL* and $\text{CTL}^*_{\mathbf{v}}$ may be seen as having the same effective expressive power.

3. Examples

In this section we present some motivating examples for RoCTL*. For more examples, see the paper that introduced RoCTL* [6], and the thesis [5].

The following example is taken from [6]. It has been adapted to only use formulas in the fragment State-RoCTL* of RoCTL* that will be introduced in Section 6, and is used here as an example of a system that can be expressed in that fragment.

An important consideration is resource management. For example, leaving a crusher idle on a mine site can be expensive so it is important to keep it supplied with ore. We now consider a simplified example where we have to keep a cat fed. Note that when specifying problems we often need for formulae to hold everywhere in the model. This can be represented with the simple CTL operator/pair of CTL* operators $\mathbf{A}\Box$ meaning "On every possible future, it is always the case that".

Example 22. We have a cat that does not eat the hour after it has eaten. If the cat bowl is empty we might forget to fill it. We must ensure that the cat never goes hungry, even if we forget to fill the cat bowl one hour. At the beginning of the first hour, the cat bowl is full. In this example the failure transitions represent failing to fill the cat bowl. We have the following atoms:

b "The cat bowl is full at the beginning of this hour"

f "This hour is feeding time"

We can translate the statements above into RoCTL* statements:

1. $\mathbf{A}\Box(f \rightarrow \mathbf{A}\bigcirc\neg f)$: (It is true everywhere that) If this hour is feeding time, the next is not.
2. $\mathbf{A}\Box((f \vee \neg b) \rightarrow \mathbf{E}\bigcirc\neg b)$: (It is true everywhere that) If it is feeding time or the cat bowl is empty, then the bowl could be empty at the next step.
3. $\mathbf{A}\Box((\neg f \wedge b) \rightarrow \mathbf{A}\bigcirc b)$: (It is true everywhere that) If the bowl is full and it is not feeding time, the bowl will be full at the beginning of the next hour.
4. $\mathbf{O}\blacktriangle\Box\,(f \rightarrow b)$: It is obligatory that, if at most one failure occurs, it is always the case that the bowl must be full at feeding time.
5. b: The cat bowl starts full.

In (2) above note that $\mathbf{E}\bigcirc\neg b$ would be equivalent to $\triangle\bigcirc\neg b$. This is because the truth of $\bigcirc\neg b$ does not depend on the truth of atoms more than one time step in the future and RoCTL* does not allow more than one failure per time step, the operator prone quantifies over deviations, and the deviation could occur now resulting in arbitrary behaviour for a single time step.

(1) and (3) could be formulated as $\mathbf{A}\Box(f \rightarrow \mathbf{A}\bigcirc\neg f)$ and $\mathbf{A}\Box((\neg f \wedge b) \rightarrow \mathbf{A}\bigcirc b)$ respectively in RoCTL* (but not State-RoCTL*). (4) Represents "If at most one failure occurs" using $\mathbf{O}$ (in a perfect future that ought to occur, that is one with no failures) and $\blacktriangle$ (even if an additional failure occurs. (5) is a trivial formula that does not use any operators, but simply states b, indicating that b is true now (but might not be true in the future). Note that failure is a property of transitions in the model, and is not explicitly referenced in the formulas. A failure is represented by following a transition that we ought not to, one that leads to a state labelled with $\mathbf{v}$. We will present below a model where failure represents forgetting to fill the cat bowl.

We can construct a model $M = (S, R, g, B)$ of this example. We let $S = \{w, w', u, u'\}$, $g(w) = \{b, f\}$, $g(w') = \{b\}$, $g(u) = \{\mathbf{v}\}$, $g(u') = \{\mathbf{v}, f\}$, $R = \{(w, w'), (w', w), (w, u), (u, u'), (u, w), (u', w'), (u', u)\}$, and B be the set of all possible fullpaths through R. Where σ is $w, w', w, w', \ldots$ we see that σ represents the case where no failures occur, and we keep the cat bowl full at all times. We see that this path also achieves the requirement (4) above, because a single failure (represented by reaching a $\mathbf{v}$ atom) may result in reaching state u where the cat bowl is not full, but that does not matter because it is not feeding time. It would take an additional failure to reach u' where it is feeding time, but the bowl is still not full.

Example 23. A simple robustness property is that a single mistake must not result in a particular error (e) state. This can be represented with $\phi = \mathbf{O}\blacktriangle\Box\neg e$.

However, when safety is paramount it is usually important to report near misses, so that they can be avoided in future. The property "A single mistake could cause ..." can be represented using the $\triangle$ operator. We can represent the property that a single deviation could (eventually) cause an error (e) as $\triangle\Diamond e$. We can then represent this "notification" requirement that if such a near miss will occur, then we will (eventually) warn (w) the user with $\triangle\Diamond e \rightarrow \Diamond w$. We can represent the notification property will hold over all possible futures with the formula $\psi = \mathbf{A}(\triangle\Diamond e \rightarrow \Diamond w)$.

Below we give an example of a description of a system that satisfies that above ϕ and ψ properties. Proving this provides a motivation for the tableau that we will define later in the paper.

Example 24. We define a system that will attempt to reach a safe state (represented by s) and warn (w) the user if the system enters an unsafe state ($\neg s$). The intuition is that if an error (e) is sufficiently serious then the system should consider any state where the error could occur within a single step to be unsafe.

1. $\mathbf{A}\Box\mathbf{O}\bigcirc s$: The system should always ensure that the system reaches a safe state by the next step.

2. $\mathbf{A}\Box(s \rightarrow \bigcirc\neg e)$: If the system is in a safe state an error e will not occur at the next step.
3. $s \wedge \neg e$: The system starts in a safe state with no error.
4. $\mathbf{A}\Box(\neg s \rightarrow \bigcirc w)$: If the system is in an unsafe state, the system will warn the user at the next step.

Note that the example above could be adapted to State-RoCTL by replacing $\bigcirc\neg e$ and $\bigcirc w$ with $\mathbf{A}\bigcirc\neg e$ and $\mathbf{A}\bigcirc w$ respectively.

We now adapt an existing example from [34].

Example 25. A sea guard system has three agents: (1) a UAV, (2) a helicopter and (3) a patrol boat. The helicopter or the UAV can monitor the area (but not both at the same time). If an unauthorised vehicle is detected then it must be intercepted by one of the three agents. The helicopter and the UAV do not both monitor the area at the same time. At most one agent will be sent to intercept any unauthorised vehicle. Neither the helicopter nor the UAV can monitor and intercept at the same time. By sending the UAV to intercept the intruder, its position is revealed. In the following we focus on formalising the constraints requiring obligation and robustness. The other constraints can be formalised using propositional logic formulae under the $\mathbf{A}\Box$ operators. We use m_1 to represent that the UAV monitors the area and m_2 to represent that the helicopter monitors the area.

The system has five requirements:

1. *The UAV must monitor the area.* Recall that in RoCTL*, obligations have to relate to the future to be non-trivial. As no stop time has been given, we interpret this as forever. Using m_1 for the UAV monitoring the area, this can be represented in RoCTL* as $\mathbf{O}\Box m_1$.
2. *If the UAV does not monitor the area, the helicopter must monitor the area.* This is a contrary-to-duty obligation, and the natural interpretation is that if the UAV fails to monitor the area, the helicopter must monitor the area. This obligation is only relevant when a failure (of the UAV) has already occurred. For the purposes of this example we do not consider the possibility that we allow a brief period to activate the helicopter after the UAV fails. Thus a natural interpretation is that when a single failure (e.g. of the UAV) has occurred, then if the UAV isn't monitoring the area the helicopter must monitor the area. This can be represented in RoCTL* as $\mathbf{O}\blacktriangle\Box(\neg m_1 \rightarrow m_2)$.
3. *If an (u)nauthorised vehicle is in the area, an agent must (i)ntercept it.* As before we can represent this as $\mathbf{O}\Box u \rightarrow i$.
4. *If all agents fail to intercept the unauthorised vehicle, an agent must (r)eport this to headquarters.* Given that there are three agents to intercept, we may wish to require that this occur even if the face of two out of the three agents failing (although if we model check this against a model where the vehicle needs to be detected by either the UAV or helicopter first, this property might not hold). We can represent this in RoCTL* with $\mathbf{O}\blacktriangle\blacktriangle\Box((u \wedge \neg i) \rightarrow r)$.

5. *The UAV must not reveal its (l)ocation.*: $\mathbf{O}\Box\neg l$.

An important consideration in computer systems is liveness, that the computer system will not freeze permanently, so if it receives an infinite number of requests it will provide an infinite number of responses. Continuing to accept requests after a failure has occurred may not always be wise. Even if a system is capable of accepting requests after a failure, a detected failure may suggest undetected safety problems or trip intrusion detection. Frequently a system that rejects requests is expected to respond with some form of denial, such as a "404: file not found" error from a web page, diagnostic information, or an apology for the inconvenience.

Example 26. Consider a system that processes a stream of requests. A single failure could leave it unable to accept requests; however, the system should at least remain live, sending "denied" responses to requests.

1. At the next time it is always both possible that a request is (r)eceived, and possible that a request is not received $\mathbf{A}\Box(\mathbf{E}\bigcirc r \wedge \mathbf{E}\bigcirc\neg r)$.
2. It ought to be the case that if an infinite number of requests are received, an infinite number of requests are (a)ccepted: $\mathbf{O}(\Box\Diamond r \rightarrow \Box\Diamond a)$
3. It ought to be robustly true that if an infinite number of requests are received, an infinite number of requests should be either accepted or (d)enied: $\mathbf{O}\blacktriangle(\Box\Diamond r \rightarrow \Box\Diamond(a \vee d))$.

Bundled logics can be more effective for representing fairness. For example, CTL* and BCTL* are complete for doubly exponential time; however, due to bundled logic's ability to encode fairness constraints in the bundle, a BCTL* tableau can be combined with a singly exponential BCTL tableau in a way that allows BCTL formulas to be interpreted over BCTL* fairness constraints of bounded size in singly exponential time [35]. With RoBCTL* we can enforce the fairness condition that an infinite number of requests are received within the bundle itself, giving a closely related specification below.

Example 27. Consider a system that processes a stream of requests:

1. (As above) At the next time it is always both possible that a request is (r)eceived, and possible that a request is not received $\mathbf{A}\Box(\mathbf{E}\bigcirc r \wedge \mathbf{E}\bigcirc\neg r)$.
2. Nevertheless, in every future an infinite number of requests will be received: $\mathbf{A}\Box\Diamond r$
3. It ought to be the case that an infinite number of requests are (a)ccepted: $\mathbf{O}\Box\Diamond a$
4. It ought to be robustly true that an infinite number of requests should be either accepted or denied: $\mathbf{O}\blacktriangle\Box\Diamond(a \vee d)$.

Note that (1) and (2) above are contradictory under CTL* and RoCTL*. From (1) we know that at each point there is a next step where $\bigcirc\neg r$ is true; in non-bundled logics $\mathbf{A}$ quantifies over every path through the structure, including the branch which at every step follows the step where $\bigcirc\neg r$ is true. Since that

branch never reaches r, $\Box\Diamond r$ is clearly false, contradicting (2). In a bundled logic, **A** only quantifies over the bundle. It is easy to show that the set of paths on which r occurs infinitely often, through some structure satisfying (1), is a bundle. It can be natural to impose the fairness constraints on the bundle itself. For example, if the requests are generated by a fair scheduler then it is a property of r itself that at each next step r may or may not be true, but r will be true infinitely often. In this case it is shorter and more natural to specify this once, rather than within each **A** and **O** formula as we would have to under non-bundled logics.

Fairness also allows a different kind of robustness to be specified. Consider the case were a request is sent at every time step, but due to packet loss is only received given some arbitrary independent probability strictly between 0 and 1. In the limit we would expect an infinite number of both packets lost and packets delivered. Showing that a system remains live under these circumstances provides a different kind of robustness to the kind provided by the $\blacktriangle$ operator. The $\blacktriangle$ operator is best used to represent a single failure, or a finite number n of failures when nested n times.

We will present a translation of State-RoCTL into CTL. Note that State-RoCTL can also be interpreted over the bundled semantics, and this could be interpreted as a translation of State-RoBCTL into BCTL. We will not discuss State-RoBCTL elsewhere in the paper, but we will discuss how to adapt Example 28 to State-RoBCTL. First note that while many fairness properties cannot be translated into CTL, $\mathbf{A}\Box\Diamond r$ is equivalent to $\mathbf{A}\Box\mathbf{A}\Diamond r$.

Example 28. Consider a system that processes a stream of requests:

1. At the next time it is always both possible that a request is (r)eceived, and possible that a request is not received $\mathbf{A}\Box(\mathbf{E}\bigcirc r \wedge \mathbf{E}\bigcirc\neg r)$. *Already State-RoBCTL/CTL.*
2. Nevertheless, In every future an infinite number of requests will be received: $\mathbf{A}\Box\Diamond r$. *Equivalent to* $\mathbf{A}\Box\mathbf{A}\Diamond r$.
3. It ought to be the case that an infinite number of requests are (a)ccepted: $\mathbf{O}\Box\Diamond a$. *Similarly equivalent to* $\mathbf{O}\Box\mathbf{O}\Diamond a$
4. It ought to be robustly true that an infinite number of requests should be either accepted or denied: $\mathbf{O}\blacktriangle\Box\Diamond(a \vee d)$. *Translating this is less trivial, but we can use*

$$\mathbf{O}\blacktriangle\Box\mathbf{O}\Diamond(a \vee d) \wedge \mathbf{O}\Box\mathbf{O}\blacktriangle\Diamond(a \vee d) \ .$$

4. A Tableau for RoBCTL*

Here we define a tableau RoBCTL*-TAB for deciding RoBCTL$^*_{\mathbf{v}}$ and thus also the restriction RoBCTL*. This tableau is an extension of Reynolds' [9] tableau for BCTL*. As with tableaux for other temporal logics (see [9] for a

fuller discussion), this tableau is not a tree, the structure of the tableau is essentially an arbitrary graph. The traditional tree-based method for constructing a tableau is to start with a single node and then build up the tableau by adding leaves. This tableau technique begins by adding all possible nodes and then acts like a sculptor, removing nodes but never adding them. Once we have finished removing nodes we find a model of the formula, provided that it is satisfiable.

An issue with branching temporal logics is that formulas are evaluated over paths rather than at worlds. As with [9], we handle this by using hues and colours. The hues correspond to paths, and are sets of formula in the closure set. The colours represent worlds and are sets of hues. The worlds are represented as the collection of paths that could start at that world. The nodes of the tableau are colours, the edges are added between any pair of nodes that satisfy a temporal successor relation. These colours can be thought of as labels, so the labels of the tableau are sets of sets of formulas.

There are two major features not present in the BCTL* tableau. We have instead used a successor function that adds enough formulas to the closure that we can handle deviations without having to distinguish paths in the Tableau. For example, the successor of "Next A" is "A". This can make the closure set large, although the size of the closure set will be elementary if the alternations between Robustly and Until are bounded.

Another feature not present in the BCTL* tableau is the ability to deal with eventualities that change over time. In BCTL* the only eventuality is of the form "A Until B", which remains unchanged until it is resolved by B occurring. In RoBCTL* we have "Eventually a path will deviate and along that path A will be hold", at the next step this becomes "... the successor of A will hold".

Definition 29. We define logical operations on sets of formulas S and T as follows:

$$(S?T) = \{\varepsilon : \exists \phi \in S,\, \exists \psi \in T \text{ s.t. } \varepsilon = (\phi ? \psi)\} \text{ where } ? \in \{\mathcal{U}, \wedge\}$$
$$?S = \{\varepsilon : \exists \phi \in S \text{ s.t. } \varepsilon = ?\phi\} \text{ where } ? \in \{\neg, \blacktriangle, \mathbf{A}, \bigcirc, \mathbf{O}\}$$

We define the abbreviations $\triangle, \mathbf{E}, \mathbf{P}, \vee, \rightarrow, \leftrightarrow$ on sets similarly.

Definition 30. We let Ξ be some normalisation function from formulas to formulas such that $\Xi(\alpha) = \Xi(\beta)$ iff α is equivalent to β under classical logic taking all subformulas with non-classical operators of highest precedence as atoms.

Note that the precise choice of Ξ is unimportant, provided the complexity is moderate compared to the doubly exponential running time of the tableau for BCTL*. Any usual normalisation such as conjunctive normal form, or picking the shortest equivalent formula would not affect the theoretical running time results.

We now wish to give an approximation $\bigcirc^{-1}$ of the inverse of the $\bigcirc$ operator. There are no past-time operators in RoBCTL*, so it is not possible to represent $\bigcirc^{-1}$ in RoBCTL* without knowing what atoms were true at the previous state.

Thus we first define $\bigcirc_a^{-1}$, where a is a set of atoms. The intention is that for any formula ϕ, fullpath σ through some structure M, then where a is the set of state formulas true at σ_0 we have $M, \sigma \vDash \phi \iff M, \sigma_{\geq 1} \vDash \bigcirc_a^{-1}(\phi)$.

Definition 31. For any set of state formulas a, we define a formula translation function $\bigcirc_a^{-1}$.

$$
\begin{aligned}
\bigcirc_a^{-1}(\phi \mathcal{U} \psi) &= \left(\bigcirc_a^{-1}(\phi) \wedge (\phi \mathcal{U} \psi)\right) \vee \bigcirc_a^{-1}(\psi) \\
\bigcirc_a^{-1}(\neg\phi) &= \neg \bigcirc_a^{-1}(\phi) \\
\bigcirc_a^{-1}(\bigcirc\phi) &= \phi \\
\bigcirc_a^{-1}(\phi \wedge \psi) &= \bigcirc_a^{-1}(\phi) \wedge \bigcirc_a^{-1}(\psi) \\
\bigcirc_a^{-1}(p) &= \begin{cases} \bot & \text{if } p \notin a \\ \top & \text{if } p \in a \end{cases} \\
\bigcirc_a^{-1}(\mathbf{A}\phi) &= \begin{cases} \bot & \text{if } \mathbf{A}\phi \notin a \\ \top & \text{if } \mathbf{A}\phi \in a \end{cases} \\
\bigcirc_a^{-1}(\mathbf{O}\phi) &= \begin{cases} \bot & \text{if } \mathbf{O}\phi \notin a \\ \top & \text{if } \mathbf{O}\phi \in a \end{cases} \\
\bigcirc_a^{-1}(\blacktriangle\phi) &= \begin{cases} \bot & \text{if } \mathbf{A}\bigcirc\mathbf{O}\Xi\left(\bigcirc_a^{-1}(\phi)\right) \notin a \\ \blacktriangle\Xi\left(\bigcirc_a^{-1}(\phi)\right) & \text{otherwise} \end{cases}
\end{aligned}
$$

To determine the required closure set, we will now define $\bigcirc^{-1}$, $\bigcirc^{-i}$ and $\bigcirc^{\star}$ in terms of $\bigcirc_a^{-1}$.

Definition 32. We define a function $\bigcirc^{-1}$ from sets of formulas to sets of formulas as follows: given a set of formulas Φ, a formula ψ is a member of $\bigcirc^{-1}(\Phi)$ iff there exists $\psi \in \Phi$ and a set of state formulas a such that $\psi \in \bigcirc_a^{-1}(\psi)$.

We now formalise and prove the idea that $\bigcirc_a^{-1}$ is the inverse of $\bigcirc$ in the lemma below.

Lemma 33. *If a is the valuation $g(\pi_0)$ of the first world of π and ϑ is a fullpath such that $\vartheta_0 = \pi_0$ then $\vartheta \vDash \phi$ iff $\vartheta_{\geq 1} \vDash \bigcirc_a^{-1}\phi$ for any formula ϕ.*

Proof. For any formula ϕ, let L_ϕ be the statement: "for all structures, and paths ϑ through that structure, we have $\vartheta \vDash \phi$ iff $\vartheta_{\geq 1} \vDash \bigcirc_a^{-1}\phi$."

It is clear that L_ϕ is true when ϕ is a state formula or a formula of the form $\bigcirc\psi$. For some pair of formulas (ϕ, ψ), say that L_ϕ and L_ψ is true, then:

($\Longrightarrow$)

1. Say that $\vartheta \vDash (\phi \mathcal{U} \psi)$,
   (a) If $\vartheta \vDash \psi$ then $\vartheta_{\geq 1} \vDash \bigcirc_a^{-1}\psi$ and so $\vartheta_{\geq 1} \vDash \bigcirc_a^{-1}(\phi \mathcal{U} \psi)$.
   (b) If $\vartheta \nvDash \psi$ then $\vartheta \vDash \phi$ and $\vartheta_{\geq 1} \vDash (\phi \mathcal{U} \psi)$. Hence $\vartheta_{\geq 1} \vDash \bigcirc_a^{-1}(\phi) \wedge (\phi \mathcal{U} \psi)$ and so $\vartheta_{\geq 1} \vDash \bigcirc_a^{-1}(\phi \mathcal{U} \psi)$.
2. Say that $\vartheta \vDash \neg\phi$. Then $\vartheta \nvDash \phi$ and so $\vartheta_{\geq 1} \nvDash \bigcirc_a^{-1}(\phi)$. Finally, $\vartheta_{\geq 1} \vDash \neg\bigcirc_a^{-1}(\phi)$.
3. Say that $\vartheta \vDash \phi \wedge \psi$. Then $\vartheta \vDash \phi$ and $\vartheta \vDash \psi$. Thus $\vartheta_{\geq 1} \vDash \bigcirc_a^{-1}\phi$ and $\vartheta_{\geq 1} \vDash \bigcirc_a^{-1}\psi$. Hence $\vartheta_{\geq 1} \vDash \bigcirc_a^{-1}(\phi) \wedge \bigcirc_a^{-1}(\psi) = \bigcirc_a^{-1}(\phi \wedge \psi)$

4. Say that $\vartheta \vDash \blacktriangle\phi$;
   (a) thus $\sigma \vDash \phi$ for any deviation σ from ϑ. Note that as a deviation, $\sigma_0 = \vartheta_0$, and hence $\sigma_{\geq 1} \vDash \bigcirc_a^{-1}(\phi)$; additionally for any path σ with $\sigma_0 = \vartheta_0$, if $\sigma_{\geq 1}$ is failure-free then σ is a deviation from ϑ and so $\vartheta \vDash \mathbf{A}\bigcirc\mathbf{O}\bigcirc_a^{-1}(\phi)$. As Ξ is a normalisation function, it follows that $\vartheta \vDash \mathbf{A}\bigcirc\mathbf{O}\Xi\left(\bigcirc_a^{-1}(\phi)\right)$; and
   (b) we will show that $\vartheta_{\geq 1} \vDash \blacktriangle\Xi\left(\bigcirc_a^{-1}(\phi)\right)$. Say that σ' is a deviation from $\vartheta_{\geq 1}$. Then from fusion closure of the set of paths B, there exists a path σ such that $\sigma_{\geq 1} = \sigma'$ and $\sigma_0 = \vartheta_0$. This path σ is a deviation from ϑ, and so $\sigma \vDash \phi$ and thus $\sigma' \vDash \bigcirc_a^{-1}(\phi)$. Hence $\vartheta_{\geq 1} \vDash \blacktriangle\bigcirc_a^{-1}(\phi)$.

($\Longleftarrow$)

1. Say that $\vartheta_{\geq 1} \vDash \bigcirc_a^{-1}(\phi\mathcal{U}\psi) = \left(\bigcirc_a^{-1}(\phi) \wedge (\phi\mathcal{U}\psi)\right) \vee \bigcirc_a^{-1}(\psi)$
   (a) If $\vartheta_{\geq 1} \vDash \bigcirc_a^{-1}(\phi) \wedge (\phi\mathcal{U}\psi)$ then $\vartheta \vDash \phi$ and $\vartheta_{\geq 1} \vDash (\phi\mathcal{U}\psi)$ so $\vartheta \vDash (\phi\mathcal{U}\psi)$.
   (b) If $\vartheta_{\geq 1} \vDash \bigcirc_a^{-1}(\psi)$ then $\vartheta \vDash \psi$ and so $\vartheta \vDash (\phi\mathcal{U}\psi)$.
2. Say that $\vartheta_{\geq 1} \vDash \neg\bigcirc_a^{-1}(\phi)$. Then $\vartheta_{\geq 1} \nvDash \bigcirc_a^{-1}(\phi)$ and so $\vartheta \nvDash \phi$. Thus $\vartheta \vDash \neg\phi$.
3. Say that $\vartheta_{\geq 1} \vDash \bigcirc_a^{-1}(\phi \wedge \psi)$. Then, from the definition of $\bigcirc_a^{-1}$ we have $\vartheta_{\geq 1} \vDash \bigcirc_a^{-1}\phi$ and $\vartheta_{\geq 1} \vDash \bigcirc_a^{-1}\psi$. Thus $\vartheta \vDash \phi$ and $\vartheta \vDash \psi$. Hence $\vartheta \vDash \phi \wedge \psi$
4. Say that $\vartheta_{\geq 1} \vDash \bigcirc_a^{-1}(\blacktriangle\phi)$. Clearly $\vartheta_{\geq 1} \nvDash \bot$, and so $\bigcirc_a^{-1}(\blacktriangle\phi) \neq \bot$. Thus, from Definition 31, we know that $\bigcirc_a^{-1}(\blacktriangle\phi) = \blacktriangle\bigcirc_a^{-1}(\phi)$ and that

$$\mathbf{A}\bigcirc\mathbf{O}\left(\bigcirc_a^{-1}(\phi)\right) \in a \ .$$

   It follows that $\vartheta \vDash \mathbf{A}\bigcirc\mathbf{O}\left(\bigcirc_a^{-1}(\phi)\right)$. Say σ is a deviation from ϑ; we will show that $\sigma_{\geq 1} \vDash \bigcirc_a^{-1}(\phi)$, and so $\sigma \vDash \phi$. Since every deviation forces ϕ it follows that $\vartheta \vDash \blacktriangle\phi$:
   (a) as $\vartheta \vDash \mathbf{A}\bigcirc\mathbf{O}\left(\bigcirc_a^{-1}(\phi)\right)$, if σ is a 0-deviation then $\sigma_{\geq 1} \vDash \bigcirc_a^{-1}(\phi)$;
   (b) if σ is an i-deviation where $i > 0$, then $\sigma_{\leq 1} = \vartheta_{\leq 1}$ and so $\sigma_{\geq 1}$ is an $(i-1)$-deviation from $\vartheta_{\geq 1}$. As $\vartheta_{\geq 1} \vDash \blacktriangle\bigcirc_a^{-1}(\phi)$, it follows that $\sigma_{\geq 1} \vDash \bigcirc_a^{-1}(\phi)$.

By induction on the length of the formula we see that the lemma holds. □

Definition 34. We define $\bigcirc^{-i}$ recursively as $\bigcirc^{-i}(\Phi) = \bigcirc^{-1}\left(\bigcirc^{1-i}(\Phi)\right)$ and $\bigcirc^{0}(\Phi) = \Phi$. Let $\bigcirc^{\star}(\Phi)$ be the normalised closure of a set of formulas Φ under $\bigcirc^{-1}$. That is, $\phi \in \bigcirc^{\star}(\Phi)$ iff there exists a non-negative integer i such that $\phi \in \Xi\left(\bigcirc^{-i}(\Phi)\right)$.

For example, $\bigcirc^{\star}(\{\bigcirc\bigcirc p\}) = \{\bigcirc\bigcirc p, \bigcirc p, p, \bot, \top\}$. Although $\bigcirc^{\star}$ and $\bigcirc^{-1}$ are finite, they can become very large. See Section 4.2 for a detailed discussion of cardinality.

Recall that $\mathbf{v}$ indicates that the previous transition was a failure, and so we can present "the next transition is a success" with $\gamma = \bigcirc\neg\mathbf{v}$.

Definition 35. Let $\gamma = \bigcirc\Box\neg\mathbf{v}$ represent the statement "this path is failure-free".

Note that γ is not a RoBCTL* formula, because it contains $\mathbf{v}$; it is a RoBCTL$^*_{\mathbf{v}}$ formula.

Definition 36. The closure $\mathbf{cl}\phi$ of the formula ϕ is defined as the smallest set that satisfies the four following requirements:

1. $\mathbf{cl}\phi \supseteq \{\phi, \gamma\}$
2. For all $\psi \in \mathbf{cl}\phi$, if $\alpha \sqsubseteq \psi$ then $\alpha \in \mathbf{cl}\phi$.
3. For all $\psi \in \mathbf{cl}\phi$, $\neg\psi \in \mathbf{cl}\phi$ or there exists α such that $\psi = \neg\alpha$ and $\alpha \in \mathbf{cl}\phi$.
4. If $\blacktriangle\psi \in \mathbf{cl}\phi$ then $\mathbf{cl}\phi \supseteq \blacktriangle\bigcirc^\star(\psi)$ and $\mathbf{cl}\phi \supseteq \mathbf{A}\bigcirc\mathbf{O}\bigcirc^\star(\psi)$.

Recall that we have defined logical operations on sets of formulas (Definition 29). Thus $\mathbf{A}\bigcirc\mathbf{O}\bigcirc^\star(\psi)$ represents the set of formulas that results when each element of $\bigcirc^\star(\psi)$ is prefixed with $\mathbf{A}\bigcirc\mathbf{O}$.

The requirement (4) above is required to ensure that the successor formulas from Definitions 32 and 31 are included in the closure set.

Definition 37 (MPC)**.** We say that $a \subseteq \mathbf{cl}\phi$ is Maximally Propositionally Consistent (MPC) iff for all $\alpha, \beta \in a$

(M1) if $\beta = \neg\alpha$ then $\beta \in a$ iff $\alpha \notin a$,

(M2) if $\alpha \wedge \beta \in \mathbf{cl}\phi$ then $(\alpha \wedge \beta) \in a$ iff ($\alpha \in a$ and $\beta \in a$)

A hue is roughly speaking a set of formulas that could hold along a single fullpath. Note though that a hue is underspecified, as $\{\mathbf{A}\bigcirc p, \mathbf{E}\bigcirc\neg p, \ldots\}$ could be a hue even though $\mathbf{A}\bigcirc p$ and $\mathbf{E}\bigcirc\neg p$ are clearly not consistent. We will need many more relations and rules to eliminate other forms of inconsistency from the tableau.

Definition 38. [Hue] A set $a \subseteq \mathbf{cl}\phi$ is a hue for ϕ iff

(H1) a is MPC;

(H2) if $\alpha\mathcal{U}\beta \in a$ then $\alpha \in a$ or $\beta \in a$;

(H3) if $\neg(\alpha\mathcal{U}\beta) \in a$ then $\beta \notin a$; and

(H4) if $\mathbf{A}\alpha \in a$ or $\blacktriangle\alpha \in a$ then $\alpha \in a$.

Note that we do not require that $\alpha \in a$ if $\mathbf{O}\alpha \in a$. As $\mathbf{O}$ is a deontic operator $\mathbf{O}\alpha \rightarrow \alpha$ is not valid.

Let H_ϕ be the set of hues of ϕ.

Definition 39. We define a function $\mathfrak{h}_\phi$ on paths such that

$$\mathfrak{h}_\phi(\pi) = \{\alpha : \alpha \in \mathbf{cl}\phi \text{ and } \pi \vDash \alpha\}$$

As H1–4 are simply properties that any set of formulas that hold along the same path must satisfy, it is clear that the following lemma holds.

Lemma 40. *From the semantics of RoBCTL*, we see that for each $\pi \in B$, $\mathfrak{h}_\phi(\pi)$ is a hue.*

Proof. (H1) Since the semantics of the $\wedge$ and $\neg$ operators in RoBCTL* come from classical logic, it is clear that $\mathfrak{h}_\phi(\pi)$ is MPC.

(H2) If $\alpha \mathcal{U} \beta \in \mathfrak{h}_\phi(\pi)$ then $\pi \vDash \alpha \mathcal{U} \beta$ and we see that either β is satisfied immediately and so $\pi \vDash \beta$ or $\pi \vDash \alpha$; hence $\alpha \in \mathfrak{h}_\phi(\pi)$ or $\beta \in \mathfrak{h}_\phi(\pi)$.

(H3) Likewise if $\neg(\alpha \mathcal{U} \beta) \in \mathfrak{h}_\phi(\pi)$ then we see that $\pi \nvDash \alpha \mathcal{U} \beta$ and so $\pi \nvDash \beta$ and so $\beta \notin \mathfrak{h}_\phi(\pi)$, demonstrating that H3 is satisfied.

(H4) If $\mathbf{A}\alpha \in \mathfrak{h}_\phi(\pi)$ then $\pi \vDash \mathbf{A}\alpha$ and so all paths starting at π_0, including π, satisfy α. Likewise if $\blacktriangle \alpha \in \mathfrak{h}_\phi(\pi)$ then $\pi \vDash \blacktriangle \alpha$ and so $\pi \vDash \alpha$. Either way $\alpha \in \mathfrak{h}_\phi(\pi)$. □

The following lemma motivates the definition of $\bigcirc_a^{-1}\phi$. We will now define a temporal successor relation r_X on hues, so called because it will satisfy $\mathfrak{h}_\phi(\pi)\, r_X\, \mathfrak{h}_\phi(\pi_{\geq 1})$ for all paths π. The definition below is similar to that found in [9], but with the additional requirements (R5) and (R6).

Definition 41. [r_X] The temporal successor r_X relation on hues is defined as follows: for all hues a, b put (a, b) in r_X iff the following conditions are satisfied:

(R1) $\bigcirc\alpha \in a$ implies $\alpha \in b$

(R2) $\neg\bigcirc\alpha \in a$ implies $\alpha \notin b$

(R3) $\alpha \mathcal{U} \beta \in a$ and $\beta \notin a$ implies $\alpha \mathcal{U} \beta \in b$

(R4) $\neg(\alpha \mathcal{U} \beta) \in a$ and $\alpha \in a$ implies $\neg(\alpha \mathcal{U} \beta) \in b$

(R5) $\blacktriangle\alpha \in a$ implies $\bigcirc_a^{-1}(\blacktriangle\alpha) \in b$

(R6) $\neg\blacktriangle\alpha \in a$ implies $\neg\bigcirc_a^{-1}(\blacktriangle\alpha) \in b$

We will now define a relation on hues $r_{\mathbf{A}}$, which informally relates hues which describe paths that can start at the same state.

Definition 42 ($r_{\mathbf{A}}$)**.** For hues a, b, we put (a, b) in $r_{\mathbf{A}}$ iff the following conditions hold:

(A1) $\mathbf{A}\alpha \in a$ iff $\mathbf{A}\alpha \in b$ and $\mathbf{O}\alpha \in a$ iff $\mathbf{O}\alpha \in b$

(A2) For all $p \in \mathbb{V}$, we have $p \in a$ iff $p \in b$

Note that if $(a, b) \in r_{\mathbf{A}}$ then for all formulas ψ we have $\bigcirc_a^{-1}(\psi) = \bigcirc_b^{-1}(\psi)$ (i.e. $\bigcirc_a^{-1} = \bigcirc_b^{-1}$). The $r_{\mathbf{A}}$ relation is used to specify which pairs of hues can exist in the same "colour"; a colour represents a set of hues for paths which could start at the same world.

Definition 43. A set of hues C is a colour of ϕ iff

(C1) for all $a, b \in C$ we have $(a, b) \in r_{\mathbf{A}}$; and

(C2) if $a \in C$ and $\neg \mathbf{A}\alpha \in a$ or $\neg \blacktriangle \alpha \in a$ then there is $b \in C$ such that $\neg\alpha \in b$; and

(C3) if $a \in C$ and $\neg \mathbf{O}\alpha \in a$ then there is $b \in C$ such that $\neg\alpha \in b$ and $\gamma \in b$; and

(C4) there exists $a \in C$ such that $\gamma \in a$

(where γ was defined in Definition 35).

Let C_ϕ be the colours of ϕ. We define a successor relation on C_ϕ as follows:

Definition 44 (R_X)**.** We define a temporal successor relation R_X on colours as follows: for all $C, D \in C_\phi$, put $(C, D) \in R_X$ iff for all $b \in D$ there exists $a \in C$ such that $(a, b) \in r_X$.

Definition 45. We define the pre-tableau (for ϕ) to be $(C_\phi, R_X, H_\phi, r_X)$.

4.1. Pruning the Pre-Tableau

The temporal successor relations ensure that each step of the pre-tableau is consistent. Note though that for any finite n, the formula $\Diamond\phi$ does not require that ϕ occur in any of the next n steps. Thus each time-step being consistent with the next is not enough to ensure that $\Diamond\phi$ is satisfied. We will use the term eventuality to informally describe formulas for which the consistency of each temporal step is not sufficient to ensure that the formula is really satisfied on the resulting model.

The following pruning procedure is based on the technique used in [9]. However, to extend this technique to RoBCTL* a new type of eventuality must be considered: $\neg\blacktriangle\psi$. This formula can be interpreted as "either $\neg\psi$ or there exists a path which eventually deviates, and $\neg\psi$ holds along that path." It is necessary to explicitly handle this eventuality. Imagine a pre-tableau with only one colour $C = \{\{\neg\blacktriangle\Box p, \Box p, p, \neg\mathbf{v}, \top\}\}$, it is clear that $(C, C) \in R_X$ so without handling eventualities of the form $\neg\blacktriangle\psi$ this colour would not be pruned. We need to ensure that eventually a path deviates on which $\Box p$ is false. The rules (1) and (2) below correspond to removal rules 1 and 2 from [9], but the rule (3) is original.

Initially, we let the set S' of colours be equal to C_ϕ. We say that a 3-tuple (C, c, α) is an instance iff $C \in S'$, c is a hue, α is a formula and $\alpha \in c \in C$. We iteratively remove colours from S' according to the following rules until no more colours can be removed:

1. Remove C from S' if we cannot find successors for every hue in C. That is, we remove C from S' if there exists a hue c in C such that for every $D \in S'$,
   (a) $(C, D) \notin R_X$, or
   (b) for every $d \in D$, the pair $(c, d) \notin r_X$.

2. An instance $(C, c, \alpha \mathcal{U} \beta)$ is directly fulfilled iff $\beta \in c$. Initially, an instance is fulfilled iff it is directly fulfilled; we iteratively mark $(C, c, \alpha \mathcal{U} \beta)$ as fulfilled iff there exists a fulfilled instance $(D, d, \alpha \mathcal{U} \beta)$ such that $(C, D) \in R_X$ and $(c, d) \in r_X$. We finish when we can no longer mark instances as fulfilled. Finally, for all instances $(C, c, \alpha \mathcal{U} \beta)$ that are not fulfilled, we remove C from S'.
3. An instance $(C, c, \neg \blacktriangle \alpha)$ is directly fulfilled iff $\mathbf{A} \bigcirc \mathbf{O} \Xi \left(\bigcirc_c^{-1} (\alpha) \right) \notin c$ or $\alpha \notin c$. Initially, an instance is fulfilled iff it is directly fulfilled; we iteratively mark $(C, c, \neg \blacktriangle \alpha)$ as fulfilled iff there exists a fulfilled instance $(D, d, \neg \blacktriangle \alpha')$ such that $(C, D) \in R_X$, $(c, d) \in r_X$ and $\blacktriangle \alpha' = \bigcirc_c^{-1} (\blacktriangle \alpha)$; we finish when we can no longer mark instances as fulfilled. Finally, for all instances $(C, c, \neg \blacktriangle \alpha)$ that are not fulfilled, we remove C from S'.

Definition 46. We say that the tableau "RoBCTL*-TAB" succeeds if there exists a hue h and colour C such that $\phi \in h \in C \in S'$ after the pruning is complete. We say that a tableau algorithm halts iff the construction process is finite.

4.2. Cardinality of the Closure Set

The complexity of the tableau is doubly exponential with respect to $|\mathbf{cl}\phi|$. To see this note that the set of hues is a subset of $2^{\mathbf{cl}\phi}$ and so the set of colours is a subset of $2^{2^{\mathbf{cl}\phi}}$. Thus constructing all the colours can be done in $2^{2^{|\mathbf{cl}\phi|}}$ time. The time required to prune a colour is polynomial in the number of colours.

We see $|\mathbf{cl}\phi|$ is linear with respect to $|\phi| \max_{\psi \sqsubseteq \phi} |\bigcirc^\star (\{\psi\})|$, from Definition 36 of $\mathbf{cl}\phi$. Thus if $|\bigcirc^\star (\{\psi\})|$ is n-exponential then the overall complexity of the tableau is $(n+2)$-exponential. In this section we will discuss the size of $|\bigcirc^\star (\{\psi\})|$.

Theorem 47. *When we require that there are no more than m pairs of alternations between $\blacktriangle$ and $\mathcal{U}$ that are not broken by an $\mathbf{A}$ (or $\mathbf{O}$) then $|\bigcirc^\star (\{\psi\})|$ is $3m$-exponential on the size of the formulas.*

We prove this by showing that we can build $|\bigcirc^\star (\{\psi\})|$ recursively: below we show that we can recurse through any number of $\blacktriangle$ operators with a singly exponential blowup until we reach a $\mathcal{U}$ operator, and we can recurse through any number of $\mathcal{U}$ operators with a doubly exponential blowup until we reach a $\blacktriangle$ operator.

Lemma 48. *Say that Φ is a set of formulas each starting with $\blacktriangle$, and ψ is a formula constructed from x instances of state formulas, y instances of $\wedge, \mathcal{U}, \bigcirc, \neg$ operators (we exclude the $\blacktriangle$ operator) and elements of Φ. Then $\bigcirc^\star (\{\psi\})$ is doubly exponential with respect to $x + y + \max_{\phi \in \Phi} \bigcirc^\star (\phi)$.*

Proof. Let $\mathbf{C}$ be a function such that $\mathbf{C}(\Phi)$ represents all normalised classical formulas with the elements of Φ as atoms. Note that a truth table on n atoms has 2^n rows, and hence there are 2^{2^n} equivalence classes on such formulas. It follows that $|\mathbf{C}(\Phi)| \leq 2^{2^{|\Phi|}}$. Let $\oint$ be a function from formulas to sets of formulas

such that $\phi \in \oint(\psi)$ iff $\bigcirc\phi \sqsubseteq \psi$ or $\phi = (\phi_1 \mathcal{U} \phi_2) \sqsubseteq \psi$. For any set of formulas Φ, we define $\oint(\Phi)$ as $\bigcup_{\phi\in\Phi} \oint(\phi)$. We see that the following statement holds:

$$\bigcirc^\star(\{\psi\}) \subseteq \mathbf{C}\left(\oint(\psi) \cup \left(\bigcup_{\blacktriangle\phi\in\Phi} \bigcirc^\star(\{\blacktriangle\phi\})\right)\right)$$

It follows that $\bigcirc^\star(\{\psi\}) \in \mathcal{O}\left(2^{2^{(x+y)+\sum_{\blacktriangle\phi\in\Phi} \bigcirc^\star(\{\blacktriangle\phi\})}}\right)$. In other words, $\bigcirc^\star(\{\psi\})$ is doubly exponential with respect to $x + y + \max_{\phi\in\Phi} \bigcirc^\star(\{\phi\})$. □

Definition 49. When considering some formula ψ, let $\#(\phi)$ be the number of times that ϕ occurs in ψ without being part of a state formula (not nested inside an **O** or **A**).

Lemma 50. *Say that Φ is a set of formulas, and ψ is a formula constructed from any number of instances of $\wedge$ and $\neg$ operators, x instances of state formulas, y instances of $\blacktriangle$ operators, and z instances of $\bigcirc$ operators (we exclude the $\mathcal{U}$ operator) and elements of Φ. Then $|\bigcirc^\star(\{\psi\})|$ is singly exponential with respect to $x + y + z + \max_{\phi\in\Phi} \bigcirc^\star(\{\phi\})$.*

Proof. Consider the set $\bigcirc^{-i}(\{\psi\})$. By inspecting the definition of $\bigcirc^{-1}$ we see that we have two choices when we reach a state formula, $\top$ and $\bot$. When we reach a $\blacktriangle$ operator, we have two choices, terminate with $\bot$ or continue to recurse. It is clear that for all $\phi \in \Phi$ and $j \in [0, \infty]$ it is the case that $\left|\bigcirc^{-j}(\{\phi\})\right| \leq |\bigcirc^\star(\{\phi\})|$. By taking the product of all these cases we get the following:

$$\left|\bigcirc^{-i}(\{\psi\})\right| \leq 2^x 2^y \prod_{\phi\in\Phi} |\bigcirc^\star(\{\phi\})|^{\#(\phi)}$$

Note that $\bigcirc^{-1}$ removes any $\bigcirc$ operator or state formula that is not inside an $\bigcirc$ or $\mathcal{U}$ operator. It is clear from induction that $\bigcirc^{-i}$ will have removed any $\bigcirc$ operator or state formula that is not inside i $\bigcirc$ operators or a $\mathcal{U}$ operator. Since ψ does not contain any $\mathcal{U}$ operator that does not form part of a $\phi \in \Phi$, it is the case that for $i > z$ we no element of $\bigcirc^{-i}(\{\psi\})$ would contain an $\bigcirc$ operator that does not form part of an element of $|\bigcirc^\star(\{\psi\})|$, and have already replaced all state formulas which do not form part of an element of $|\bigcirc^\star(\{\psi\})|$ with either $\top$ or $\bot$. It follows that

$$\left|\bigcup_{i>z} \bigcirc^{-i}(\{\psi\})\right| \leq 2^x 2^y \prod_{\phi\in\Phi} |\bigcirc^\star(\{\phi\})|^{\#(\phi)}.$$

Since $\bigcirc^\star(\{\psi\}) = \bigcup_{i\geq 0} \Xi\bigcirc^{-i}(\{\psi\})$ and $\bigcirc^{-0}(\{\psi\}) = 1$,

$$|\bigcirc^\star(\{\psi\})| = \left|\bigcup_{i\geq 0} \bigcirc^{-i}(\{\psi\})\right| \leq 1 + (1+z)\, 2^x 2^y \prod_{\phi\in\Phi} |\bigcirc^\star(\{\phi\})|^{\#(\phi)}.$$

□

Corollary 51. *The tableau provides a 3-exponential decision procedure for the fragments of RoCTL* and RoBCTL* without Until.*

Proof. Clear from Lemmas 50 and 17. □

Definition 52. Let $\blacktriangle^n$ be a sequence $\blacktriangle\blacktriangle\ldots\blacktriangle$ of n instances of the $\blacktriangle$ operator.

We see that also $\bigcirc^\star(\{\blacktriangle^n\phi\}) = \{\blacktriangle x\colon\ x \in \bigcirc^\star(\phi)\} \cup \{\bot\}$. The $\blacktriangle^n$ operator is interesting as it represents the statement "even with n additional failures" and a significant factor in the design of the $\blacktriangle$ was to provide a simple unimodal operator that could represent this statement. However, in the QCTL* based decision procedure for RoCTL* [32] the $\blacktriangle^n$ operator involves a non-elementary blowup in the complexity. By comparison, in this tableau the complexity is independent of n in $\blacktriangle^n$.

$$\begin{aligned}
\psi &::= \phi \wedge \phi \mid \neg\phi \mid \mathbf{A}\psi \mid \mathbf{O}\psi \mid \bigcirc\psi \mid \blacktriangle\psi \mid \alpha\mathcal{U}\alpha \\
\alpha &::= \alpha \wedge \alpha \mid \neg\alpha \mid \mathbf{A}\psi \mid \mathbf{O}\psi \mid \bigcirc\alpha \mid \blacktriangle\beta \mid \alpha\mathcal{U}\alpha \text{ (inside } \mathcal{U}) \\
\beta &::= \beta \wedge \beta \mid \neg\beta \mid \mathbf{A}\psi \mid \mathbf{O}\psi \mid \bigcirc\beta \mid \blacktriangle\beta \text{ (inside } \mathcal{U}, \blacktriangle)
\end{aligned}$$

We now show that when we restrict ourselves to formulas without a $\mathcal{U}$ operator nested within a $\blacktriangle$ operator nested within a $\mathcal{U}$ operator (unbroken by an **A** or **O**), the tableau running time become elementary.

Corollary 53. $|\bigcirc^\star(\{\psi\})|$ *is 4-exponential with respect to* ψ *and the running time of the tableau is at worst 6-exponential when we restrict input as described by the following BNF:*

$$\begin{aligned}
\psi &::= \phi \wedge \phi \mid \neg\phi \mid \mathbf{A}\psi \mid \mathbf{O}\psi \mid \bigcirc\psi \mid \blacktriangle\psi \mid \alpha\mathcal{U}\alpha \\
\alpha &::= \alpha \wedge \alpha \mid \neg\alpha \mid \mathbf{A}\psi \mid \mathbf{O}\psi \mid \bigcirc\alpha \mid \blacktriangle\beta \mid \alpha\mathcal{U}\alpha \textit{ (inside } \mathcal{U}) \\
\beta &::= \beta \wedge \beta \mid \neg\beta \mid \mathbf{A}\psi \mid \mathbf{O}\psi \mid \bigcirc\beta \mid \blacktriangle\beta \textit{ (inside } \mathcal{U}, \blacktriangle)
\end{aligned}$$

Proof. Let Φ_1 be the set of formulas that do not contain any $\mathcal{U}$ operators. From Lemma 50 we see that $|\bigcirc^\star(\{\phi\})|$ is singly exponential in $|\phi|$ for $\phi \in \Phi_1$. Let Φ_2 be the set of formulas that consist only of elements of Φ_1 and operators other than $\mathcal{U}$. From Lemma 48 we see that $|\bigcirc^\star(\{\phi\})|$ is 3-exponential in $|\phi|$ for $\phi \in \Phi_2$. Let Φ_3 be the set of formulas that consist only of elements of Φ_2 and operators other than $\blacktriangle$. From Lemma 48 we see that $|\bigcirc^\star(\{\phi\})|$ is 4-exponential in $|\phi|$ for $\phi \in \Phi_3$. We see that Φ_3 is precisely the set of formulas that do not contain a $\mathcal{U}$ operator nested within a $\blacktriangle$ operator nested within a $\mathcal{U}$ operator, and furthermore that $\bigcirc^\star(\{\mathbf{A}\phi\}) = \bigcirc^\star(\{\mathbf{O}\phi\}) = \{\top, \bot\}$, and so these nestings would not pose a problem if broken by an **O** or **A**. Finally since the tableau is 2-exponential in the size of the closure set the result follows. □

The potential real world uses for Ro(B)CTL* suggested by examples such as those in Section 3, do not require multiple alternations between $\mathcal{U}$ and $\blacktriangle$ and so they would be elementary to decide (at worst 6-exponential). Also clearly

$|\bigcirc^\star(\{\psi\})|$ is $\mathcal{O}(1)$ for fixed ψ or for ψ of fixed length. As such $\mathbf{cl}\phi$ is linear on $|\phi|$ when the length of the subformulas with $\blacktriangle$ as the operator of highest precedence is bounded, and complexity is 2-exponential like CTL*. This indicates that we can introduce some (fixed) RoBCTL* properties into a set of systems specified in BCTL* without affecting the overall complexity.

4.3. Soundness

In this section we prove the following lemma.

Lemma 54. *RoBCTL*-TAB is sound, that is, if it succeeds on ϕ then ϕ is satisfiable in RoBCTL**

Say that RoBCTL*-TAB finishes with the set S' of colours. Then we define a RoBCTL-structure (S, R, g, B) as follows: the transition frame (S, R) is simply (S', R_X), and the valuation $g(C)$ of a world/colour C contains an atom p iff the hues in C contain p. We now define the set of bundled paths B.

We will now define a thread. Informally, a thread is a sequence which demonstrates the existence of a path which satisfies some hue h_0 in some colour c_0. To do this it provides a sequence of hues and colours in r_X and R_X. To understand the need for restrictions on this sequence, recall firstly that $(\alpha \mathcal{U} \beta)$ is not satisfied unless we eventually reach a β. Secondly $\neg\blacktriangle\alpha$ is equivalent to $\triangle\neg\alpha$, and is only satisfied if $\neg\alpha$ is true on the current path, or if we can eventually deviate onto a path where $\neg\alpha$ is satisfied. This second requirement requires a more complex definition, as the eventuality that we need to satisfy changes. For example, if $\triangle\neg\bigcirc\bigcirc p$ is in h_0 then this eventuality becomes $\triangle\neg\bigcirc p$ in the subsequent hue h_1. This is because if there is a deviation needs to $\neg\bigcirc p$ to ensure $\neg\bigcirc\bigcirc p$ is satisfied in the previous step.

Definition 55. We call an ω-sequence $\langle (c_0, h_0), (c_1, h_1), \ldots \rangle$ a thread through S' iff for all $i \geq 0$: each $c_i \in S'$, each $h_i \in c_i$, each $(c_i, c_{i+1}) \in R_X$, each $(h_i, h_{i+1}) \in r_X$. We say that this is a fulfilling thread iff for all $i \geq 0$

1. For all formulas of the form $(\alpha \mathcal{U} \beta)$ in h_i, there exists $j \geq i$ such that $\beta \in h_j$
2. For all formulas of the form $\neg\blacktriangle\alpha$ in h_i, $\neg\alpha \in h_i$ or there exists $j \geq i$ such that

$$\bigcirc_{h_j}^{-1} \bigcirc_{h_{j-1}}^{-1} \bigcirc_{h_{j-2}}^{-1} \ldots \bigcirc_{h_i}^{-1} (\blacktriangle\alpha) = \bot$$

We include a fullpath $\sigma = \langle c_0, c_1, \ldots \rangle$ in B iff there exists a fulfilling thread $\langle (c_0, h_0), (c_1, h_1), \ldots \rangle$, and we say that this thread justifies σ being in B.

Recall from Section 4.1 that an instance $(C, c, \neg\blacktriangle\alpha)$ is directly fulfilled iff $\mathbf{A}\bigcirc\mathbf{O}\Xi\left(\bigcirc_c^{-1}(\alpha)\right) \notin c$ or $\alpha \notin c$; from Definition 31 that

$$\bigcirc_a^{-1}(\blacktriangle\phi) = \begin{cases} \bot & \text{if } \mathbf{A}\bigcirc\mathbf{O}\Xi\left(\bigcirc_a^{-1}(\phi)\right) \notin a \\ \blacktriangle\Xi\left(\bigcirc_a^{-1}(\phi)\right) & \text{otherwise} \end{cases}.$$

The following lemma follows naturally from these definitions, though some details need to be considered.

Lemma 56. *Requirement (2) above is equivalent to the statement: there exists $j \geq i$ such that $\left(c_j, h_j, \neg\bigcirc^{-1}_{h_{j-1}}\bigcirc^{-1}_{h_{j-2}}\ldots\bigcirc^{-1}_{h_i}(\blacktriangle\alpha)\right)$ is directly fulfilled.*

Proof. ($\Longrightarrow$) Say requirement (2) holds.

Case 1. $\neg\alpha \in h_i$. Then $(c_i, h_i, \neg(\blacktriangle\alpha))$ is directly fulfilled.

Case 2. There exists $j \geq i$ such that

$$\bigcirc^{-1}_{h_j}\bigcirc^{-1}_{h_{j-1}}\bigcirc^{-1}_{h_{j-2}}\ldots\bigcirc^{-1}_{h_i}(\blacktriangle\alpha) = \bot$$

Say Without Loss of Generality (WLOG) that j is as small as possible, e.g.

$$\bigcirc^{-1}_{h_{j-1}}\bigcirc^{-1}_{h_{j-2}}\ldots\bigcirc^{-1}_{h_i}(\blacktriangle\alpha) \neq \bot$$

Then there exists β such that

$$\blacktriangle\beta = \bigcirc^{-1}_{h_{j-1}}\bigcirc^{-1}_{h_{j-2}}\ldots\bigcirc^{-1}_{h_i}(\blacktriangle\alpha)$$

and

$$\bigcirc^{-1}_{h_j}\blacktriangle\beta = \bot$$

Thus $\mathbf{A}\bigcirc\mathbf{O}\Xi\left(\bigcirc^{-1}_{h_j}(\beta)\right) \notin h_j$ and so $(c_j, h_j, \neg\blacktriangle\beta)$ is directly fulfilled. Hence

$$\left(c_j, h_j, \neg\bigcirc^{-1}_{h_{j-1}}\bigcirc^{-1}_{h_{j-2}}\ldots\bigcirc^{-1}_{h_i}(\blacktriangle\alpha)\right)$$

is directly fulfilled.

($\Longleftarrow$) Say there exists $j \geq i$ such that $\left(c_j, h_j, \neg\bigcirc^{-1}_{h_{j-1}}\bigcirc^{-1}_{h_{j-2}}\ldots\bigcirc^{-1}_{h_i}(\blacktriangle\alpha)\right)$ is directly fulfilled. Hence, there exists β such that $\blacktriangle\beta = \bigcirc^{-1}_{h_{j-1}}\bigcirc^{-1}_{h_{j-2}}\ldots\bigcirc^{-1}_{h_i}(\blacktriangle\alpha)$.

Case 1. $\mathbf{A}\bigcirc\mathbf{O}\Xi\left(\bigcirc^{-1}_{h_j}(\beta)\right) \notin h_j$. Then $\bigcirc^{-1}_{h_j}(\blacktriangle\beta) = \bot$ and so

$$\bigcirc^{-1}_{h_j}\bigcirc^{-1}_{h_{j-1}}\bigcirc^{-1}_{h_{j-2}}\ldots\bigcirc^{-1}_{h_i}(\blacktriangle\alpha) = \bot$$

Case 2. $\neg\beta \in h_j$. From Corollary 57 below, $\beta \in h_j$ iff $\alpha \in h_i$. Hence $\neg\alpha \in h_i$. □

Corollary 57. *Say that there exists β such that*

$$\blacktriangle\beta = \bigcirc^{-1}_{h_{j-1}}\bigcirc^{-1}_{h_{j-2}}\ldots\bigcirc^{-1}_{h_i}(\blacktriangle\alpha)\ .$$

Where $\blacktriangle\alpha \in \mathbf{cl}\phi$. Then $\beta \in h_j$ iff $\alpha \in h_i$.

From the definition of $\mathbf{cl}\phi$ we see that $\blacktriangle\alpha \in \mathbf{cl}\phi \implies \blacktriangle\beta \in \mathbf{cl}\phi$. Hence we see that the above corollary follows from Lemma 56 and induction. We may likewise use induction to show that $\sigma_{\geq i} \vDash \alpha$ iff $\sigma_{\geq j} \vDash \bigcirc^{-1}_{h_{j-1}}\bigcirc^{-1}_{h_{j-2}}\ldots\bigcirc^{-1}_{h_i}(\blacktriangle\alpha)$.

We will show that B is a bundle by showing that B is suffix closed, fusion closed and non-empty (recall Definition 4).

Lemma 58. *B is suffix closed.*

Proof. Say $\mu = \langle (c_0, h_0), (c_1, h_1), \ldots \rangle$ justifies $\sigma \in B$. Clearly if there is an eventuality at σ_m which is fulfilled at σ_n then $n \geq m$. Thus for any suffix $\sigma_{\geq j}$ then if σ_n is not on $\sigma_{\geq j}$ then σ_m is not on $\sigma_{\geq j}$ either. Hence we see that for all $j \geq 0$, $\mu_{\geq j}$ justifies $\sigma_{\geq j} \in B$. □

Lemma 59. *B is fusion closed.*

Proof. Say that σ, π are in B and $\sigma_0 = \pi_1$. We will show below that

$$\langle \pi_0, \sigma_0, \sigma_1, \ldots \rangle \in B \ .$$

The general case where $\sigma_0 = \pi_j$ follows from prefix closure and induction.

As $\sigma \in B$, there is a fulfilling thread $\mu = \langle (\sigma_0, h_1), (\sigma_1, h_2), \ldots \rangle$. As $(\pi_0, \pi_1) \in R_X$, we can choose h_0 from π_0 such that $(h_0, h_1) \in r_X$.

If $\alpha \mathcal{U} \beta \in h_0$, then $\beta \in h_0$ or $\alpha \mathcal{U} \beta \in h_1$. As μ is fulfilling, if $\alpha \mathcal{U} \beta \in h_1$ then there exists $j \geq 1$ such that $\beta \in h_j$.

If $\neg \blacktriangle \alpha \in h_0$ then

1. $\neg \alpha \in h_0$ or $\neg \mathbf{A} \bigcirc \mathbf{O} \Xi \left(\bigcirc_a^{-1} (\alpha) \right) \in h_0$; or
2. $\neg \blacktriangle \Xi \left(\bigcirc_{h_0}^{-1} (\alpha) \right) \in h_1$. (From (R6) and Definition 31)

If (1) then the eventuality $\neg \blacktriangle \alpha$ is directly fulfilled. Otherwise, from Lemma 56, there exists β and j such that

$$\begin{aligned} \blacktriangle \beta =& \bigcirc_{h_{j-1}}^{-1} \bigcirc_{h_{j-2}}^{-1} \ldots \bigcirc_{h_1}^{-1} \left(\blacktriangle \Xi \left(\bigcirc_{h_0}^{-1} (\alpha) \right) \right) \\ =& \bigcirc_{h_{j-1}}^{-1} \bigcirc_{h_{j-2}}^{-1} \ldots \bigcirc_{h_0}^{-1} (\blacktriangle \alpha) \end{aligned}$$

and $\mathbf{A} \bigcirc \mathbf{O} \Xi \left(\bigcirc_{h_j} (\beta) \right) \notin h_j$ or $\beta \notin h_j$. Thus the eventuality $\neg \blacktriangle \alpha$ is fulfilled by h_j. □

We now show that we can construct a fulfilling thread in the lemma below. As with Reynolds' 2007 BCTL* tableau [9] the intuition is that we can ensure that all eventualities are satisfied by iteratively satisfy the oldest eventuality first. Since the number of eventualities is finite (indeed no more in size than the closure set), and the number of steps to fulfil each eventuality is finite, all of the eventualities will be fulfilled in a finite number of steps. Thus, even though an eventually may reoccur an infinite number of times in the thread, this strategy ensures it will always be satisfied again later in the thread.

Lemma 60. *If $a \in c \in S'$ then there is a fulfilling thread*

$$\mu = \langle (c_0, h_0), (c_1, h_1), \ldots \rangle$$

such that $h_0 = a$ and $c_0 = c$. Thus $\sigma = \langle c_0, c_1, c_2, \ldots \rangle \in B$.

Proof. Say we have chosen the first n elements of μ and $0 \leq i \leq n$. We say that an eventuality $\alpha \mathcal{U} \beta \in h_i$ is unfulfilled iff for all $j \leq i \leq n$ the formula $\beta \notin h_j$. We say that an eventuality $\neg \blacktriangle \phi \in h_i$ is unfulfilled iff for all $j \leq i \leq n$ the instance

$$\left(c_i, h_i, \neg \bigcirc^{-1}_{h_{j-1}} \bigcirc^{-1}_{h_{j-2}} \dots \bigcirc^{-1}_{h_i} (\blacktriangle \alpha)\right)$$

is not directly fulfilled.
For the lowest i such that there exists an unfulfilled eventuality in h_i we fulfil this eventuality as follows:

Case 1. If no such i exists, we choose (c_{n+1}, h_{n+1}) such that $(c_n, c_{n+1}) \in R_X$ and $(h_n, h_{n+1}) \in r_X$.

Case 2. If the eventuality is of the form $\alpha \mathcal{U} \beta$, then there must exist $\alpha \mathcal{U} \beta \in h_n$. Due to the pruning rule, for some j there must exist a sequence of instances

$$(c_n, h_n, \alpha \mathcal{U} \beta), (c_{n+1}, h_{n+1}, \alpha \mathcal{U} \beta), \dots, (c_j, h_j, \alpha \mathcal{U} \beta)$$

such that the final instance is directly fulfilled ($\beta \in h_j$), and each other instance is fulfilled by the next instance in the chain. Having now chosen μ up to (c_j, h_j) with $\beta \in h_j$, the eventuality $\alpha \mathcal{U} \beta \in h_i$ is now fulfilled.

Case 3. If the eventuality is of the form $\neg \blacktriangle \alpha$, then there must exist $\neg \blacktriangle \phi \in h_n$ where

$$\blacktriangle \phi = \bigcirc^{-1}_{h_{n-1}} \bigcirc^{-1}_{h_{n-2}} \dots \bigcirc^{-1}_{h_i} (\blacktriangle \alpha).$$

Due to the pruning rule, for some j there must exist a sequence of instances

$$(c_n, h_n, \neg \blacktriangle \phi), \left(c_{n+1}, h_{n+1}, \neg \bigcirc^{-1}_{h_n} \blacktriangle \phi\right), \dots, \left(c_j, h_j, \neg \bigcirc^{-1}_{h_{j-1}} \bigcirc^{-1}_{h_{j-2}} \cdots \bigcirc^{-1}_{h_n} \blacktriangle \phi\right)$$

such that the final instance is directly fulfilled, and each other instance is fulfilled by the next instance in the chain. Having now chosen μ up to (c_j, h_j), the eventuality $\neg \blacktriangle \alpha \in h_i$ is now fulfilled, as

$$\neg \bigcirc^{-1}_{h_{j-1}} \bigcirc^{-1}_{h_{j-2}} \cdots \bigcirc^{-1}_{h_n} \blacktriangle \phi = \neg \bigcirc^{-1}_{h_{j-1}} \bigcirc^{-1}_{h_{j-2}} \cdots \bigcirc^{-1}_{h_n} \bigcirc^{-1}_{h_{n-1}} \bigcirc^{-1}_{h_{n-2}} \dots \bigcirc^{-1}_{h_i} (\blacktriangle \alpha),$$

and

$$\left(c_j, h_j, \neg \bigcirc^{-1}_{h_{j-1}} \bigcirc^{-1}_{h_{j-2}} \cdots \bigcirc^{-1}_{h_i} (\blacktriangle \alpha)\right),$$

is directly fulfilled. □

In the following lemma we formalise and prove the intuition that a formula is true along a thread iff it is in the first hue.

Lemma 61. *For all α in $\mathbf{cl}\phi$, for all threads $\mu = \langle (c_0, h_0), (c_1, h_1), \dots \rangle$ justifying $\sigma = \langle c_0, c_1, \dots \rangle$ we have*

$$(S, R, g, B), \sigma \vDash \alpha \text{ iff } \alpha \in h_0$$

Proof. We will prove this using induction. Let L_ψ be the statement: this lemma holds for the case where $\alpha = \psi$.

First note that L_ψ holds by definition when ψ is an atom. Assume that L_ψ holds for all ψ in $\mathbf{cl}\phi$ where $|\psi| \leq n$ and the number of nested $\blacktriangle$ operators in ψ is less than m. We can show that L_ψ holds for any ψ in $\mathbf{cl}\phi$ where $|\psi| \leq n+1$ and the number of nested $\blacktriangle$ operators is less than $m+1$. We will show below the case where ψ is of the form $\blacktriangle\beta$, the other forms are left as an exercise for the reader (or see [9]).

($\Longrightarrow$) Suppose by contradiction that $\blacktriangle\beta \notin h_0$ so that $\neg\blacktriangle\beta \in h_0$. If $\neg\beta \in h_0$ then $\sigma \vDash \neg\beta$, which gives us a contradiction. From (R6) we know that $\neg\bigcirc_{h_0}^{-1}(\blacktriangle\beta)$ is in h_1. Note that $\bigcirc_{h_0}^{-1}(\blacktriangle\beta)$ is either $\bot$ or of the form $\blacktriangle\psi$. If $\bigcirc_{h_0}^{-1}(\blacktriangle\beta)$ is of the form $\blacktriangle\psi$, we likewise see that $\bigcirc_{h_1}^{-1}\bigcirc_{h_0}^{-1}(\blacktriangle\beta) \in h_2$ and so on; however, due to pruning rule 3 this cannot go on forever, there must exist finite i such that $\bigcirc_{h_i}^{-1}\bigcirc_{h_{i-1}}^{-1}\bigcirc_{h_{i-2}}^{-1}\ldots\bigcirc_{h_0}^{-1}(\blacktriangle\beta) = \bot$. It is now easy to use Lemma 33 and induction to show that $\sigma \vDash \blacktriangle\beta$ iff $\sigma \vDash \bot$, which again gives us a contradiction.

($\Longleftarrow$) Suppose by contradiction that $(S, R, g, B), \sigma \nvDash \blacktriangle\beta$. If $\sigma \nvDash \beta$ then $\beta \notin h_0$, so for some i there must exist an i-deviation π from σ such that $\pi \nvDash \beta$. We can show that $\pi_{\geq i} \nvDash \bigcirc_{\pi_{i-1}}^{-1}\cdots\bigcirc_{\pi_1}^{-1}\bigcirc_{\pi_0}^{-1}(\beta)$ and so

$$\Xi\bigcirc_{\pi_{i-1}}^{-1}\cdots\Xi\bigcirc_{\pi_1}^{-1}\Xi\bigcirc_{\pi_0}^{-1}(\beta) \notin \pi_i.$$

Thus

$$\bigcirc_{\pi_{i-1}}^{-1}\cdots\bigcirc_{\pi_1}^{-1}\bigcirc_{\pi_0}^{-1}(\blacktriangle\beta) = \blacktriangle\Xi\bigcirc_{\pi_{i-1}}^{-1}\cdots\Xi\bigcirc_{\pi_1}^{-1}\Xi\bigcirc_{\pi_0}^{-1}(\beta) \notin \pi_i.$$

It follows that $\blacktriangle\beta \notin \pi_0 = \sigma_0$. □

4.4. Completeness

In this section we prove the following lemma.

Lemma 62. *RoBCTL*-TAB is complete, that is, if ϕ is satisfiable in RoBCTL$^*_{\mathbf{v}}$ then RoBCTL*-TAB halts and succeeds on ϕ.*

The tableau is finite so RoBCTL*-TAB will halt. Say that ϕ is satisfiable. Then there exists a RoBCTL$_{\mathbf{v}}$ structure (S, R, g, B) and path π^0 in B such that $\pi^0 \vDash \phi$. We will define a translation ρ_ϕ from worlds to colours, and show that for each world w in S, the colour $\rho_\phi(w)$ will not be pruned from the tableau. Hence S' will be non-empty when RoBCTL*-TAB halts, and so RoBCTL*-TAB will succeed.

We will now define a function from worlds to colours. This definition uses the function from paths to hues defined in Definition 39.

Definition 63. We define a function ρ_ϕ on worlds to sets of hues:

$$\rho_\phi(w) = \{\mathfrak{h}_\phi(\pi)\colon\ \pi \in B \text{ and } \pi_0 = w\}\ .$$

Lemma 64. *For each $w \in S$, $\rho_\phi(w)$ is a colour.*

Proof. Obvious, as C1–C4 are trivial consequences of the semantics of RoBCTL*. □

Case 1 and 2 below are similar to the corresponding proof in [9]. Case 3 is original. In each case it intuitive that the pruning rule would not find an inconsistency since $\rho_\phi(w)$ is derived from an actual model and so cannot be inconsistent unless some other consistent node has already pruned.

Lemma 65. *For each $w \in S$, $\rho_\phi(w)$ is never removed from S'.*

Proof. Suppose for contradiction that $\rho_\phi(w)$ is removed. Without loss of generality, choose w such that $\rho_\phi(w)$ is the first such node to be pruned.

Case 1. $\rho_\phi(w)$ is removed from S' by rule 1 as we cannot find successors for every hue in $\rho_\phi(w)$. Then there exists a hue $c \in \rho_\phi(w)$ such that for every $D \in S'$, $(\rho_\phi(w), D) \notin R_X$, or for every $d \in D$, the pair $(c,d) \notin r_X$. Since $c \in \rho_\phi(w)$ there must exist a fullpath $\pi \in B$ such that $\pi_0 = w$. It is easy to verify that $(\rho_\phi(\pi_0), \rho_\phi(\pi_1)) \in R_X$, that $\mathfrak{h}_\phi(\pi_{\geq 1}) \in \rho_\phi(\pi_1)$ and that $(c, \mathfrak{h}_\phi(\pi_{\geq 1})) \in r_X$.

Case 2. $\rho_\phi(w)$ is removed from S' by rule 2 as it contains an unfulfillable eventuality of the form $\alpha \mathcal{U} \beta$.

Thus there is $a \in \rho_\phi(w)$ such that $\alpha \mathcal{U} \beta \in a$, but $\beta \notin a$ and there is no sequence of instances

$$\langle (c_0, h_0, \alpha \mathcal{U} \beta), (c_1, h_1, \alpha \mathcal{U} \beta), \ldots, (c_n, h_n, \alpha \mathcal{U} \beta) \rangle$$

with $n \geq 0$ such that $c_0 = \rho_\phi(w)$, $h_0 = a$, $\alpha_0 = \alpha$ each $h_i \in c_i \in S_0$, each $(c_i, c_{i+1}) \in R_X$, each $(h_i, h_{i+1}) \in r_X$ and with $\beta \in h_n$.

As $a \in \rho_\phi(w)$, there exists π in B such that $\pi_0 = w$ and $\mathfrak{h}_\phi(\pi) = a$. Hence $\pi \vDash \alpha \mathcal{U} \beta$ and so there exists a non-negative integer i such that $\pi_{\geq i} \vDash \beta$. Hence the instance $(\rho_\phi(\pi_i), \mathfrak{h}_\phi(\pi_{\geq i}), \alpha \mathcal{U} \beta)$ is directly fulfilled, and so such a sequence of instances does exist:

$$\langle (\rho_\phi(\pi_0), \mathfrak{h}_\phi(\pi), \alpha \mathcal{U} \beta), (\rho_\phi(\pi_1), \mathfrak{h}_\phi(\pi_{\geq 1}), \alpha \mathcal{U} \beta), \ldots, (\rho_\phi(\pi_i), \mathfrak{h}_\phi(\pi_{\geq i}), \alpha \mathcal{U} \beta) \rangle.$$

By contradiction, $\rho_\phi(w)$ is not removed.

Case 3. $\rho_\phi(w)$ is removed from S' by rule 3 as it contains an unfulfillable eventuality of the form $\neg \blacktriangle \alpha$. Thus there is $a \in \rho_\phi(w)$ such that $\neg \blacktriangle \alpha \in a$, but $\alpha \in a$ and there is no sequence of instances

$$\langle (c_0, h_0, \neg \blacktriangle \alpha_0), (c_1, h_1, \neg \blacktriangle \alpha_1), \ldots, (c_n, h_n, \neg \blacktriangle \alpha_n) \rangle$$

with $n \geq 0$ such that $c_0 = \rho_\phi(w)$, $h_0 = a$, $\alpha_0 = \alpha$ each $h_i \in c_i \in S_0$, each $(c_i, c_{i+1}) \in R_X$, each $(h_i, h_{i+1}) \in r_X$, each $\blacktriangle \alpha_{i+1} = \bigcirc_{h_i}^{-1}(\blacktriangle \alpha_i)$ and $(c_n, h_n, \neg \blacktriangle \alpha_n)$ being directly fulfilled.

Since $a \in \rho_\phi(w)$ there exists π with $a = \mathfrak{h}_\phi(\pi), \pi \in B, \pi_0 = w$. Now $\neg \blacktriangle \alpha \in a = \mathfrak{h}_\phi(\pi)$ so $\pi \nvDash \blacktriangle \alpha$. If $\pi \nvDash \alpha$ then $\neg \alpha \in a$ and $(c_0, h_0, \neg \blacktriangle \alpha_0)$ is directly fulfilled. If $\pi \vDash \alpha$ then for some i there exists an i-deviation $\sigma \in B$ from π such that $\sigma \nvDash \alpha$. For purposes of contradiction we will now show how that such a sequence of instances does exist with $n = i$.

From the definition of 31 we see that for each $j \leq n$ we have $\alpha_{j+1} = \bigcirc_{h_j}^{-1}(\alpha_j)$, or $\alpha_{j+1} = \bot$. Since $\pi \vDash \alpha$ we know $\alpha_{j+1} \neq \neg\bot$ (from Lemma 33). From $\sigma \nvDash \alpha$, Lemma 33, $\alpha_{j+1} = \bigcirc_{h_j}^{-1}(\alpha_j)$ and induction it follows that $\sigma_{\geq i} \nvDash \alpha_i$. By definition of an i-deviation we see that $\sigma_{\geq i+1}$ is failure-free. As $\pi_i = \sigma_i$ and $\sigma_{\geq i} \nvDash \alpha_i$ it follows that $\sigma_{\geq i+1} \nvDash \bigcirc_{h_i}^{-1}\alpha_i$ and so $\pi_{\geq i} \nvDash \mathbf{A}\bigcirc\mathbf{O}\bigcirc_{h_i}^{-1}\alpha_i$. Hence, $\mathbf{A}\bigcirc\mathbf{O}\Xi(\alpha_{i+1}) \notin \mathfrak{h}_\phi(\pi_{\geq i})$, and so $(\rho_\phi(\pi_i), \mathfrak{h}_\phi(\pi_{\geq i}), \neg\blacktriangle\alpha_i)$ is directly fulfilled. Thus such a sequence does exist:

$$\langle(\rho_\phi(w), \mathfrak{h}_\phi(\pi), \neg\blacktriangle\alpha), (\rho_\phi(\pi_1), \mathfrak{h}_\phi(\pi_{\geq 1}), \neg\blacktriangle\alpha_1) \ldots (\rho_\phi(\pi_i), \mathfrak{h}_\phi(\pi_{\geq i}), \neg\blacktriangle\alpha_i)\rangle$$

By contradiction, $\rho_\phi(w)$ is not removed. □

4.5. Example of Tableau

Here we will consider the tableau for the trivial case where $\phi = p$. Our closure must contain $\bigcirc\Box\neg\mathbf{v}$ and negations of subformulas, so our closure $\mathbf{cl}\phi$ is $\{\neg\bigcirc\Box\neg\mathbf{v}, \bigcirc\Box\neg\mathbf{v}, \neg\Box\neg\mathbf{v}, \Box\neg\mathbf{v}, \neg\mathbf{v}, \mathbf{v}, p, \neg p\}$.

From (M1) know that for any hue h each member α of the closure set, either α or $\neg a$ is in α (but not both). We will begin to enumerate the hues.

$$\begin{aligned}
h_1 &= \{\neg\bigcirc\Box\neg\mathbf{v}, \neg\Box\neg\mathbf{v}, \neg\mathbf{v}, \neg p\} \\
h_2 &= \{\neg\bigcirc\Box\neg\mathbf{v}, \neg\Box\neg\mathbf{v}, \neg\mathbf{v}, p\} \\
h_3 &= \{\neg\bigcirc\Box\neg\mathbf{v}, \neg\Box\neg\mathbf{v}, \mathbf{v}, \neg p\} \\
h_4 &= \{\neg\bigcirc\Box\neg\mathbf{v}, \neg\Box\neg\mathbf{v}, \mathbf{v}, p\} \\
h_5 &= \{\neg\bigcirc\Box\neg\mathbf{v}, \Box\neg\mathbf{v}, \neg\mathbf{v}, \neg p\} \\
h_6 &= \{\neg\bigcirc\Box\neg\mathbf{v}, \Box\neg\mathbf{v}, \neg\mathbf{v}, p\}
\end{aligned}$$

The next rows are:

$$\{\neg\bigcirc\Box\neg\mathbf{v}, \Box\neg\mathbf{v}, \mathbf{v}, \neg p\}$$
$$\{\neg\bigcirc\Box\neg\mathbf{v}, \Box\neg\mathbf{v}, \mathbf{v}, p\}.$$

However, these are not hues. Recall that $\Box\neg\mathbf{v}$ is an abbreviation for $\neg(\top\mathcal{U}\mathbf{v})$, and so these rows contradict H3. Hues 7–12 are similar to hues 1–6, but with $\neg\bigcirc\Box\neg\mathbf{v}$ replaced with $\bigcirc\Box\neg\mathbf{v}$. For example,

$$h_{12} = \{\bigcirc\Box\neg\mathbf{v}, \Box\neg\mathbf{v}, \neg\mathbf{v}, p\} \ .$$

Since we have 12 hues, we have possible 2^{12} sets of hues to consider when constructing the colours. For space reasons we will not explicitly consider all these possibilities. Even for the trivial input formula p, to directly apply the algorithm is non-trivial for a human. In this case only one colour is required to show that p is satisfiable: $\{h_{12}\}$. This colour loops back onto itself.

It is well known that unwinding a graph tableau into a tree-like tableau gives much greater efficiency. It has been shown how to do such a tree unwinding for tableau based on hues and colours such as this one. In the hue-colour tree tableau for BCTL* [12] it is not required that the hues be maximally

propositionally consistent. A hue without either α or $\neg\alpha$ represents paths that may or may not contain α. This allows the BCTL* tableau for ϕ to begin with a single node: $\{\emptyset, \{\phi\}\}$ and only construct the nodes reachable from that node. This greatly improves real world performance, but is not necessary for theoretical results. In the case of RoBCTL* we could implement a similar tableau beginning with $\{\emptyset, \{\phi\}, \{\bigcirc\Box\neg\mathbf{v}\}\}$.

5. Pair-RoCTL

Recall that reasoning about CTL* is harder than CTL. For example, satisfiability checking for CTL* is double-exponentially complete [36, 37, 19], whereas this problem for CTL can be decided in singly exponential time. For this reason it is of interest to find a CTL-like restriction of RoCTL*. One possibility is to require (as with CTL) that every temporal operator be paired with a path quantifier.

Recall Definition 10 of Pair-RoCTL. Pair-RoCTL is a syntactic fragment of RoCTL* that is similar to CTL in the sense that every path operator is paired with a path quantifier and vice-versa.

We will present a satisfiability preserving translation τ from CTL* into Pair-RoCTL. This translation has some special atoms $y_{\mathbf{A}\psi}$ and $y_{\mathbf{E}\psi}$. Note that these special atoms are not required for translating LTL into Pair-RoCTL. As model checking LTL is as hard as for CTL*, it would be sufficient to provide a translation from the LTL formulas, as it is a well known result that LTL is as hard to model check as CTL* [38, 39], both are PSPACE-Complete. However, satisfiability checking is considerably harder for CTL* than LTL, as CTL* is 2-EXPTIME hard [19] (and complete [37]) for double exponential time while satisfiability checking LTL is complete for polynomial space [39].

The intuition behind the following translation is that we force the CTL* formula to be evaluated over paths consisting solely of failures. These paths cannot be deviations, as deviations have a failure-free suffix. Thus $\triangle\phi$ will be satisfied iff ϕ is satisfied, and so we can pair each $\bigcirc$ or $\mathcal{U}$ operator with a $\triangle$ without changing whether the formula is satisfied.

Requiring that $\Box\bigcirc\mathbf{v}$ hold along a path would ensure that the path is fully failing; however, $\mathbf{v}$ is a special atom which does not occur in RoCTL* formulas. We instead create an atom f such that f is only true when the last transition was a failure,[1] and the restriction of the structure to subset of the states where f is true form a CTL model. We then use f in place of $\mathbf{v}$.

When comparing RoCTL* and CTL* we will only consider RoCTL$_\mathbf{v}$ structures. This means that

1. we can check that we are being evaluated over a fully failing path (one that satisfies $\Box f$) by use of $\triangle\Box f$; and

[1] If we are using RoCTL$_\mathbf{v}$ structures then this means $\bigcirc f$ will be true only where $\bigcirc\mathbf{v}$ is true. Note that as Pair-RoCTL is a restriction of RoCTL*, Pair-RoCTL formulas do not include the atom $\mathbf{v}$.

2. we can represent any linear temporal operator (e.g. $\Diamond$) in Pair-RoCTL by prefixing it with $\triangle$ (e.g. we translate $\Diamond$ into $\triangle\Diamond$)

We will now define a translation τ from CTL* to Pair-RoCTL. The intention is that $\tau(\phi)$ is satisfiable iff ϕ is satisfiable.

Definition 66. We define a function τ from CTL* formulas to Pair-RoCTL formulas as follows:

Let ϕ be a CTL* formula in Negation Normal Form (NNF). We will begin by defining κ_f,τ_y, **prev**, $\tau_{\mathbf{prev}}$, τ_1 and τ_p which we will use to define a translation function τ. The Pair-RoCTL formula κ_f ensures that the f atom remains false once it becomes false, that f is only true if the last transition was a failure transition and that the subset of the worlds S that satisfy f true is serial.

$$\begin{aligned}\kappa_f = \mathbf{A}\Box(\neg f \rightarrow \mathbf{A}\bigcirc\neg f)\wedge \\ \mathbf{A}\Box\mathbf{O}\bigcirc\neg f \wedge \mathbf{A}\Box(f \rightarrow \mathbf{E}\bigcirc f)\ .\end{aligned}$$

The Pair-RoCTL formula $\tau_y(\phi)$, defined below, is used to encode the state-formulas of an arbitrary CTL* formula ϕ into atoms. It ensures that each atom of the form $y_{\mathbf{A}\psi}$ is only true at those states that satisfy $\mathbf{A}\psi$. Likewise each atom of the form $y_{\mathbf{E}\psi}$ is only true at those states that satisfy $\mathbf{E}\psi$. Note that we do not require that $y_{\mathbf{A}\psi}$ be true at states that satisfy $\mathbf{A}\psi$. This is because $y_{\mathbf{A}\psi}$ occurs only positively in $\tau_1(\phi)$, which will be defined below, and so making $y_{\mathbf{A}\psi}$ false will never make $\tau_1(\phi)$ to be true. Thus a requirement that $\mathbf{A}\psi \implies y_{\mathbf{A}\psi}$ would be redundant when testing for satisfiability.

$$\begin{aligned}\tau_y(\phi) = \bigwedge_{\mathbf{A}\psi\sqsubseteq\phi} \mathbf{A}\Box\left((y_{\mathbf{A}\psi} \wedge (\triangle\Box f)) \rightarrow \tau_1(\psi)\right)\wedge \\ \bigwedge_{\mathbf{E}\psi\sqsubseteq\phi} \mathbf{A}\Box\left(y_{\mathbf{E}\psi} \rightarrow \mathbf{E}\bigcirc\mathbf{prev}\left(\tau_1\left(\psi\right)\right)\right)\ .\end{aligned}$$

We now define the function **prev** from Pair-RoCTL formulas to Pair-RoCTL formulas with the intention that $\bigcirc\mathbf{prev}(\psi) \leftrightarrow \psi$ on all paths through our structure. By $[p/p']$ we denote replacing the atom p with p', and by $[\forall p \sqsubseteq \psi : p/p']$ we denote similarly replacing all atoms.

$$\mathbf{prev}(\psi) = \psi[\forall p \sqsubseteq \psi :\ p/p']$$

The translation above replaces each occurrence of p with p', relying on p' being true exactly when p was true at the last state. We define the function $\tau_{\mathbf{prev}}$ below from Pair-RoCTL formulas to Pair-RoCTL formulas for the purpose of ensuring that this holds for each atom of the form p' that occurs in some formula ψ. We denote that α is a subformula of β by $\alpha \sqsubseteq \beta$.

$$\tau_{\mathbf{prev}}(\psi) = \mathbf{A}\Box\left(\bigwedge_{p'\sqsubseteq\psi}(p \rightarrow \mathbf{A}\bigcirc p') \wedge \bigwedge_{p'\sqsubseteq\psi}(\neg p \rightarrow \mathbf{A}\bigcirc\neg p')\right)\ .$$

We can now define τ_p, which adds all the new atoms required by combining κ_f τ_y and $\tau_{\mathbf{prev}}$.

$$\tau_p(\phi) = \kappa_f \wedge \tau_y(\phi) \wedge \tau_{\mathbf{prev}}(\tau_y(\phi)) \ .$$

We now define the τ_1 translation. This is the core of the τ translation, but it depends on atoms introduced by the τ_p translation for correctness.

$$\begin{aligned}
\tau_1(\mathbf{A}\psi) &= y_{\mathbf{A}\psi} \wedge \triangle\Box f \\
\tau_1(\mathbf{E}\psi) &= y_{\mathbf{E}\psi} \wedge \triangle\Box f \\
\tau_1(\bigcirc\psi) &= (\triangle\bigcirc\tau_1(\psi)) \\
\tau_1(\psi * \phi) &= \triangle[\tau_1(\psi) * \tau_1(\phi)], * \in \mathcal{U}, \overset{\neg}{\mathcal{U}} \\
\tau_1(p) &= (p \wedge \triangle\Box f) \\
\tau_1(\neg p) &= (\neg p \wedge \triangle\Box f)
\end{aligned}$$

Finally we define $\tau(\phi)$ itself as follows:

$$\tau(\phi) = \tau_1(\phi) \wedge \tau_p(\phi) \ .$$

5.1. Translating a CTL Model into a Pair-RoCTL Model*

Recall Definition 15 of CTL-structures. Given a CTL-structure $M = (S, R, g)$, we construct a RoCTL-structure $M^{\mathfrak{R}} = \left(S^{\mathfrak{R}}, R^{\mathfrak{R}}, g^{\mathfrak{R}}\right)$ from M as follows:

- we add a new "success" world s so that $S^{\mathfrak{R}} = S \cup \{s\}$
- the accessibility relation $R^{\mathfrak{R}}$ is the least relation that satisfies $R^{\mathfrak{R}} \supseteq R$ and $\langle w, s\rangle \in R^{\mathfrak{R}}$ for all $w \in S^{\mathfrak{R}}$
- the valuation $g^{\mathfrak{R}}$ satisfies the following:
  - for every atom p in the original formula ϕ and $w \in S$ we have $p \in g^{\mathfrak{R}}(w)$ iff $p \in g(w)$
  - the failure atom f and violation atom $\mathbf{v}$ is true at every world except the success world. Formally, $f \in g^{\mathfrak{R}}(w)$ iff $w \neq s$ and $\mathbf{v} \in g^{\mathfrak{R}}(w)$ iff $w \neq s$
  - for every atom of the form y_ψ in $\tau_y(\phi)$ and every world $w \in S$ it is the case that $y_\psi \in g^{\mathfrak{R}}(w)$ iff $M, w \vDash \psi$.

Note that we need the f atom as the $\mathbf{v}$ atom cannot explicitly appear in any RoCTL* or Pair-RoCTL formula.

We will now define a function h to add the p' atoms into the model.

Definition 67. We define a function h from $\text{RoCTL}_{\mathbf{v}}$ structures to $\text{RoCTL}_{\mathbf{v}}$ structures such that for any $\text{RoCTL}_{\mathbf{v}}$ structure $M = (S, R, g)$ we have $h(M) = \left(S^h, R^h, g^h\right)$ where:

1. $S^h = R$, so every world in $h(M)$ is of the form (w, v) where w and v are in S (that is worlds of M). Being in world (w, v) means roughly "we are currently at world v but were at world w previously"
2. for any pair of worlds (w, v) and (x, y) in S^h we have $(w, v) R^h (x, y)$ iff $x = v$ and $(x, y) \in R$
3. for all $p \in \mathbb{V}$, it is the case that $p \in g^h(\langle w, v\rangle) \iff p \in g(v)$ and $p' \in g^h(\langle w, v\rangle) \iff p \in g(w)$

We will use $\langle ?, w\rangle$ to represent $\langle v, w\rangle$ for some arbitrary v, when we do not care about truth values of the p' atoms at this world. For convenience we extend the definition of h such that $h(\sigma) = \langle ?, \sigma_0\rangle, \langle \sigma_0, \sigma_1\rangle, \langle \sigma_1, \sigma_2\rangle \ldots$, and $h(M, \sigma) = h(M), h(\sigma)$.

Lemma 68. *For all RoCTL$_{\mathbf{v}}$ structures M, fullpaths σ through M and RoCTL* formulas ϕ that do not contain atoms of the form p' we have $M, \sigma \vDash \phi \iff h(M, \sigma) \vDash \phi \iff h(M, \sigma) \vDash \bigcirc\mathbf{prev}(\phi)$. If each atom p' is true exactly when p was true at the previous world (or $M, \sigma \vDash \tau_{\mathbf{prev}}(\phi)$) for some formula ϕ then $h(M, \sigma) \vDash \phi \iff h(M, \sigma) \vDash \bigcirc\mathbf{prev}(\phi)$.*

Proof. We have shown that RoCTL$^*_{\mathbf{v}}$ is bisimulation invariant in [6], and so it is clear that $M, \sigma \vDash \phi \iff h(M, \sigma) \vDash \phi$. It is easy to see that $h(M, \sigma) \vDash p \iff h(M, \sigma) \vDash \bigcirc p'$, or in other words that $h(M, \sigma) \vDash \psi \iff h(M, \sigma) \vDash \bigcirc\mathbf{prev}(\psi)$ for ψ of length 1. Thus it is clear from induction on the length of formula and the semantics of RoCTL* that $h(M, \sigma) \vDash \psi \iff h(M, \sigma) \vDash \bigcirc\mathbf{prev}(\psi)$ for ψ of any length. □

Definition 69. Given a model M for an Pair-RoCTL formula $\tau(\phi)$ we construct a model M^C for the CTL* formula ϕ as follows: remove all worlds where f is false.

5.2. Proof of Correctness

Without loss of generality we can assume that each structure has a world w_0 such that every other world is reachable from that world, so for example $M, w_0 \vDash \mathbf{A}\Box p$ means that p is true at all worlds in M.

When we use "$(\neg)$" in a sentence this indicates that the sentence remains true if all occurrences of $(\neg)$ are replaced with a $\neg$ or if all occurrences of $(\neg)$ are simply removed. This is similar to how the $\pm$ operator is frequently used.

Lemma 70. *For any structure M that satisfies $M, w_0 \vDash \mathbf{A}\Box\mathbf{O}\bigcirc(\neg f)$, it is the case that $M, \pi \vDash \triangle\Box f \implies M, \pi \vDash \Box f$ and $M, \pi \vDash \triangle\tau_1(\psi) \implies M, \pi \vDash \tau_1(\psi)$ for all paths π through M and formulas ψ.*

Proof. As $M, \pi \vDash \triangle\Box f$ either $M, \pi \vDash \Box f$ or there exists a deviation σ from π that satisfies $\Box f$. Any deviation σ from π has a failure-free suffix. That is there exists i such that $\sigma_{\geq i}$ is failure-free. As $M, w_0 \vDash \mathbf{A}\Box\mathbf{O}\bigcirc\neg f$, it is the case that $M, \sigma_{\geq i} \vDash \mathbf{O}\bigcirc\neg f$, and as $\sigma_{\geq i}$ is failure-free, $M, \sigma_{\geq i+1} \vDash \neg f$ and so $M, \sigma \nvDash \Box f$. Hence, by contradiction, $M, \pi \vDash \Box f$.

As any deviation π from σ has a failure-free suffix and $M, \sigma \not\models \Box f$, we see that $M, \sigma \not\models ((\neg) p \wedge \triangle \Box f)$ for any $p \in \mathbb{V}$. Thus we see that $M, \sigma \not\models \tau_1(\psi)$ for any ψ of length 1. It is easy to see by recursion on length of ψ that $M, \sigma \not\models \tau_1(\psi)$ for ψ of any length. It follows that $M, \pi \models \triangle \tau_1(\psi) \implies M, \pi \models \tau_1(\psi)$. $\square$

For the next lemma, recall that s was the success world defined at the beginning of Section 5.1.

Lemma 71. *Say that $M^{\mathfrak{R}}, \sigma \models \tau_1(\psi)$ for some path σ through $M^{\mathfrak{R}}$, then $s \neq \sigma_i$ for any non-negative integer i.*

Proof. For any fullpath π such that $M^{\mathfrak{R}}, \pi \not\models \triangle \Box f$ we see that for ψ of the form $\mathbf{A}\theta$, $\mathbf{E}\theta$, p or $\neg p$ it is the case that $M^{\mathfrak{R}}, \pi \not\models \tau_1(\psi)$. If there exists an integer i such that $\sigma_i = s$ we see that $M^{\mathfrak{R}}, \sigma_j \not\models \Box f$ for any $j \in \mathbb{N}$, and hence by Lemma 70 above we have $M^{\mathfrak{R}}, \sigma_j \not\models \triangle \Box f$. By induction, we see that $M^{\mathfrak{R}}, \sigma \not\models \tau_1(\psi)$, for any formulas ψ. $\square$

We will now show that the translation preserves truth.

Lemma 72. *For any CTL* formula ϕ, CTL-structure M and fullpath π through M we have $h\left(M^{\mathfrak{R}}, \pi\right) \models \tau(\phi)$ if $M, \pi \models \phi$.*

Proof. Let $H(\psi)$ be the statement: for any CTL-structure M and fullpath π through M we have $h\left(M^{\mathfrak{R}}, \pi\right) \models \tau(\psi)$ if $M, \pi \models \psi$. We will now show $H(\psi)$ is true for all ψ such that $|\psi| = 1$, that is all ψ that consist of a single atom.

It is easy to see that for every path σ through M we have $M^{\mathfrak{R}}, \sigma \models \triangle \Box f$ and so for every $p \in \mathbb{V}$, it is the case that $M, \sigma \models (\neg) p \implies M^{\mathfrak{R}}, \sigma \models ((\neg) p \wedge \triangle \Box f)$. Thus for all formulas ψ that consist of a single atom it is the case that $M, \sigma \models \psi \implies M^{\mathfrak{R}}, \sigma \models \tau_1(\psi)$. Say that $M, \sigma \models \psi \implies h\left(M^{\mathfrak{R}}, \sigma\right) \models \tau_1(\psi)$ for all ψ of length less than some integer n. Now we assume that $H(\psi)$ is true for all ψ such that $|\psi| \leq n$ for some positive integer n, and will prove that $H(\psi)$ holds for any ψ such that $|\psi| = n + 1$.

$\psi = \alpha \mathcal{U} \beta$: For all σ it is the case that $M, \sigma \models \alpha \implies M^{\mathfrak{R}}, \sigma \models \tau_1(\alpha)$ and $M, \sigma \models \beta \implies M^{\mathfrak{R}}, \sigma \models \tau_1(\beta)$. Say that $M, \sigma \models \alpha \mathcal{U} \beta$ then there exists an integer i such that $M, \sigma_{\geq i} \models \beta$ and for all integers j less than i we have $M, \sigma_{\geq j} \models \alpha$. Thus $M^{\mathfrak{R}}, \sigma_{\geq i} \models \tau_1(\beta)$ and for all j less than i we have $M^{\mathfrak{R}}, \sigma_{\geq j} \models \tau_1(\alpha)$. Hence $M^{\mathfrak{R}}, \sigma \models \tau_1(\alpha) \mathcal{U} \tau_1(\beta)$, so $M^{\mathfrak{R}}, \sigma \models \triangle(\tau_1(\alpha) \mathcal{U} \tau_1(\beta)) = \tau_1(\psi)$.

$\psi = \alpha \bar{\mathcal{U}} \beta$: Say that $M, \sigma \models \alpha \bar{\mathcal{U}} \beta$. Then $M, \sigma \models \neg(\neg\alpha \mathcal{U} \neg\beta)$, and so for all $i \in \mathbb{N}$ either $M, \sigma_{\geq i} \not\models \neg\beta$ or there exists $j < i$ such that $M, \sigma_{\geq j} \not\models \neg\alpha$. Thus $M^{\mathfrak{R}}, \sigma_{\geq i} \models \tau_1(\beta)$ or there exists $j < i$ such that $M^{\mathfrak{R}}, \sigma_{\geq j} \models \tau_1(\alpha)$ and so $M^{\mathfrak{R}}, \sigma \models \neg(\tau_1(\neg\alpha) \mathcal{U}(\neg\beta))$. Hence

$$M^{\mathfrak{R}}, \sigma \models \triangle\neg(\tau_1(\neg\alpha) \mathcal{U}(\neg\beta)) \quad = \quad \tau_1(\alpha \bar{\mathcal{U}} \beta).$$

$\psi = \bigcirc\alpha$: if $M, \sigma \models \bigcirc\alpha$ then $M, \sigma_{\geq 1} \models \alpha$. Thus $M^{\mathfrak{R}}, \sigma_{\geq 1} \models \tau_1(\alpha)$, and so $M^{\mathfrak{R}}, \sigma \models \bigcirc\tau_1(\alpha) = \tau_1(\psi)$.

$\psi = \mathbf{E}\alpha$: By definition, $M, \sigma \vDash \mathbf{E}\alpha$ iff $M^{\mathfrak{R}}, \sigma \vDash y_{\mathbf{E}\alpha}$. As σ is a path through M, it does not contain the success state and so $M^{\mathfrak{R}}, \sigma \vDash \Box f$ and so if $M, \sigma \vDash \mathbf{E}\alpha$ then $M^{\mathfrak{R}}, \sigma \vDash y_{\mathbf{E}\alpha} \wedge \triangle\Box f = \tau_1(\psi)$.

$\psi = \mathbf{A}\alpha$: As above.

It is now clear from induction that $M^{\mathfrak{R}}, \pi \vDash \tau_1(\phi)$. We see that case for $\mathbf{A}\alpha$ and $\mathbf{E}\alpha$ above are trivial. The complexity was taken outside τ_1 in the form:

$$\bigwedge_{\mathbf{A}\psi \sqsubseteq \phi} \mathbf{A}\Box((y_{\mathbf{A}\psi} \wedge (\triangle\Box f)) \rightarrow \tau_1(\psi))$$

$$\bigwedge_{\mathbf{E}\psi \sqsubseteq \phi} \mathbf{A}\Box\left(y_{\mathbf{E}\psi} \rightarrow \mathbf{E}\bigcirc\mathbf{prev}\left(\tau_1\left(\psi\right)\right)\right)$$

Now $y_{\mathbf{E}\psi}$ occurs exactly on those σ_0 where $M, \sigma \vDash \mathbf{E}\psi$. Thus $M^{\mathfrak{R}}, \sigma \vDash \mathbf{E}\tau_1(\psi)$ and $h\left(M^{\mathfrak{R}}, \sigma\right) \vDash \mathbf{E}\tau_1(\psi)$, thus $h\left(M^{\mathfrak{R}}, \sigma\right) \vDash \mathbf{E}\bigcirc\mathbf{prev}\left(\tau_1\left(\psi\right)\right)$. Likewise $y_{\mathbf{A}\psi}$ occurs exactly on those σ_0 where $M, \sigma \vDash \mathbf{A}\psi$, that is, for every path σ' through M such that $\sigma'_0 = \sigma_0$ it is the case that $M, \sigma \vDash \psi$ and so $M^{\mathfrak{R}}, \sigma \vDash \tau_1(\psi)$ and $M^{\mathfrak{R}}, \sigma \vDash (y_{\mathbf{A}\phi} \wedge (\triangle\Box f)) \rightarrow \tau_1(\psi)$. If the path σ is not through M, it contains the success state s, and so $\sigma \nvDash \triangle\Box f$ and so again $M^{\mathfrak{R}}, \sigma \vDash (y_{\mathbf{A}\phi} \wedge (\triangle\Box f)) \rightarrow \tau_1(\psi)$.

It is now easy to show that the way $M^{\mathfrak{R}}$ is constructed ensures that $h\left(M^{\mathfrak{R}}, \sigma\right) \vDash \tau_p(\phi)$, and since $h\left(M^{\mathfrak{R}}, \sigma\right) \vDash \tau_1(\phi)$, it follows that $h\left(M^{\mathfrak{R}}, \sigma\right) \vDash \tau(\phi)$.

□

Given any CTL-structure M we can construct the RoCTL$_{\mathbf{v}}$ structures $M^{\mathfrak{R}}$ and $h\left(M^{\mathfrak{R}}\right)$. From Lemma 72 above it is clear that $\tau(\phi)$ is satisfiable if ϕ is satisfiable. We have not yet shown that for any RoCTL$_{\mathbf{v}}$ structure we can construct an equivalent CTL-structure. We will now deal with constructing a CTL-structure from an arbitrary RoCTL$_{\mathbf{v}}$ structure; from Lemma 73 below it is clear that ϕ is satisfiable, if $\tau(\phi)$ is satisfiable. By combining these we get Theorem 74, with which we will conclude this section.

Lemma 73. *For any RoCTL$_{\mathbf{v}}$ structure M, If $M, w_0 \vDash \tau(\phi)$ then $M^C, w_0 \vDash \phi$.*

Proof. Recall that M^C is the translation of M defined in Definition 69; that is the structure M with the worlds that do not satisfy f removed.

As $M \vDash \tau(\phi)$, clearly $M \vDash \tau_p(\phi)$. Since $M \vDash \tau_p(\phi)$ we see that $M, w_0 \vDash \mathbf{A}\Box\mathbf{O}\bigcirc(\neg f)$ from Lemma 70 and again it is the case that $M, \pi \vDash \triangle\Box f \implies M, \pi \vDash \Box f$ and $M, \pi \vDash \triangle\tau_1(\psi) \implies M, \pi \vDash \tau_1(\psi)$ for all CTL* formulas ψ and fullpaths π through M.

Say that for all paths σ through M and all CTL* formulas ψ with $|\psi| \leq n$ for some integer n, it is the case that $M, \sigma \vDash \tau_1(\psi) \implies M, \sigma \vDash \psi$. Now consider the case where $|\psi| = n + 1$. Where ψ is of the form $(\neg)\, p$, $\alpha\vec{\mathcal{U}}\beta$, $\alpha\mathcal{U}\beta$ or $\bigcirc\alpha$ we see that $M, \sigma \vDash \tau_1(\psi) \implies M, \sigma \vDash \psi$ using the same arguments made in the previous lemma.

Now say ψ is of the form $\mathbf{A}\theta$, and say $M, \sigma \vDash \tau_1(\psi) = y_{\mathbf{A}\theta} \wedge \triangle\Box f$. From τ_y we know that $\mathbf{A}\Box((y_{\mathbf{A}\theta} \wedge (\triangle\Box f)) \rightarrow \tau_1(\theta))$. Consider a path π through M^C such that $\pi_0 = \sigma_0$, i.e. $\pi \in \delta^\omega(\sigma_0)$. We know that $M^C, \pi_0 \vDash \Box f$ as f is true at every world in M^C, so likewise $M, \pi \vDash \Box f$ and $M, \pi \vDash \triangle\Box f$. From $\tau_1(\phi)$ we know that $M, \pi \vDash y_{\mathbf{A}\theta}$ and from τ_y we know that $\mathbf{A}\Box((y_{\mathbf{A}\theta} \wedge (\triangle\Box f)) \rightarrow \tau_1(\theta))$, hence $M, \pi \vDash \tau_1(\theta)$. Since $|\theta| \leq n$ it follows that $M^C, \pi \vDash \theta$, for all $\pi \in \delta^\omega(\sigma_0)$. Thus $M^C, \sigma \vDash \mathbf{A}\theta$.

Say ψ is of the form $\mathbf{E}\theta$ and $M, \sigma \vDash \tau_1(\psi) = y_{\mathbf{E}\theta} \wedge \triangle\Box f$. As $M, \sigma \vDash y_{\mathbf{E}\theta}$, from τ_y we know that $M, \sigma \vDash \mathbf{E}\bigcirc\mathbf{prev}(\tau_1(\theta))$. As $M, \sigma \vDash \tau_{\mathbf{prev}}(\tau_y(\phi))$, we see that $M, \sigma \vDash \mathbf{E}(\tau_1(\theta))$. Finally, since $|\theta| \leq n$ we know that $M, \sigma \vDash \mathbf{E}\theta$. □

Theorem 74. *$\tau(\phi)$ is satisfiable in Pair-RoCTL* iff ϕ is satisfiable in CTL*.

5.3. Model Checking

Given a formula ϕ, and a model checking procedure for Pair-RoCTL we can compute the RoCTL$_{\mathbf{v}}$ structure $M^{\mathfrak{R}}$ from the CTL-structure M as follows.

First we add the f atom as above. Then, for subformulas α of the form $\mathbf{A}\psi$ (or $\mathbf{E}\psi$) we perform the following, starting with the shortest subformula α. For each world t in $M^{\mathfrak{R}}$ we pick an arbitrary world $\langle s, t\rangle$ in $h\left(M^{\mathfrak{R}}\right)$ and model check $\mathbf{E}\bigcirc\mathbf{prev}(\tau_1(\psi))$, if this formula holds at $\langle s, t\rangle$ we add y_ϕ to the valuation of the world t.

We now have the model $h\left(M^{\mathfrak{R}}\right)$, such that $h\left(M^{\mathfrak{R}}\right) \vDash \tau(\phi)$ iff $M \vDash \phi$. Thus where $m = |\phi|$ and n is the number of worlds in M we can easily reduce the problem of model checking a CTL* formula to mn model checking problems of Pair-RoCTL formula of length $\mathcal{O}(m)$ on models with no more than n^2 worlds. As the model checking algorithm for CTL* is PSPACE-complete [24], the model checking problem for Pair-RoCTL is PSPACE-hard.

5.4. Expressivity

Previously we have given a translation from CTL* to Pair-RoCTL that preserves satisfiability. However, the translation was not truth-preserving, and a satisfiability preserving translation tells us nothing about the expressivity of the language. After all, we can easily construct a satisfiability preserving translation from CTL* to $\top$ and $\bot$. Here we will briefly define a translation that is truth-preserving, but has a singly exponential blowup in the size of the resulting formulas.

We now present a translation function τ from CTL* formulas to Pair-RoCTL formulas such that for all structures M and paths σ it is the case that $M^{\mathfrak{R}}, \sigma \vDash \tau(\phi)$ iff $M, \sigma \vDash \phi$ is satisfiable; where $M^{\mathfrak{R}}$ is RoCTL$_{\mathbf{v}}$ structure generated from M as by adding a success world s and f, $\mathbf{v}$ atoms as in Section 5.1. Unlike Section 5.1 we need not at any atoms of the form $y_{\mathbf{A}\psi}$, $y_{\mathbf{E}\psi}$ or p' to $M^{\mathfrak{R}}$.

It is well known that for we can split a formula ϕ into state formulas and formulas that are true at the next step, that is into the form:

$$\bigvee_i \alpha_i \wedge \bigcirc\psi_i \ ,$$

where α is a list of state formulas and ψ is a list of formulas. Informally, we will define $\tau_1(\mathbf{A}\phi)$ and $\tau_1(\mathbf{E}\phi)$ in terms the above split as follows:

$$\begin{aligned}\tau_1(\mathbf{A}\phi) &= \bigvee_i \tau_1(\alpha_i) \wedge \mathbf{A}\bigcirc\tau_1(\psi_i) \\ \tau_1(\mathbf{E}\phi) &= \bigvee_i \tau_1(\alpha_i) \wedge \mathbf{E}\bigcirc\tau_1(\psi_i) \ .\end{aligned}$$

We will now define such a split so that we can provide a precise definition of this τ_1.

Recall Definition 31 of $\bigcirc_a^{-1}$, a set of formula translation functions which has the following property:

Lemma 75. *Let σ be an arbitrary path, ϕ an arbitrary formula and again let Φ be the set of all state subformulas of ϕ. Then $\sigma \vDash \phi$ iff $\sigma_{\geq 1} \vDash \bigcirc_a^{-1}(\phi)$.*

This lemma (and its proof) is very similar to Lemma 33.

Definition 76. Let Φ be the set of all state subformulas of ϕ. Then we define a function **split** from formulas to formulas such that for any formula ϕ

$$\mathbf{split}(\phi) = \bigvee_{a \in 2^\Phi} \left(\left(\bigwedge_{\psi \in a} \psi \right) \wedge \left(\bigwedge_{\psi \in (\Phi - a)} \neg\psi \right) \wedge \bigcirc\bigcirc_a^{-1}(\phi) \right)$$

Lemma 77. *Let σ be an arbitrary path, ϕ an arbitrary formula. Then $\sigma \vDash \phi$ iff $\sigma \vDash \boldsymbol{split}(\phi)$.*

Proof. We see that exactly one clause of the form

$$\left(\bigwedge_{\psi \in a} \psi \right) \wedge \left(\bigwedge_{\psi \in (\Phi - a)} \neg\psi \right)$$

will hold along σ: the clause where for each $\psi \in \Phi$ we have $\psi \in a$ iff $M, \sigma \vDash \psi$. Thus $M, \sigma \vDash \mathbf{split}(\phi)$ iff $M, \sigma \vDash \bigcirc_a^{-1}(\phi)$ for this a, and so from Lemma 75 we know that $M, \sigma \vDash \phi$. □

The formula translation function τ_1 is defined as follows:

$$\begin{aligned}\tau_1(\mathbf{A}\phi) &= \bigvee_{a \in 2^\Phi} \left(\left(\bigwedge_{\psi \in a} \psi \right) \wedge \left(\bigwedge_{\psi \in (\Phi - a)} \neg\psi \right) \wedge \mathbf{A}\bigcirc\tau_1\left(\bigcirc_a^{-1}(\phi)\right) \right) \\ \tau_1(\mathbf{E}\phi) &= \bigvee_{a \in 2^\Phi} \left(\left(\bigwedge_{\psi \in a} \psi \right) \wedge \left(\bigwedge_{\psi \in (\Phi - a)} \neg\psi \right) \wedge \mathbf{E}\bigcirc\tau_1\left(\bigcirc_a^{-1}(\phi)\right) \right) \ ,\end{aligned}$$

for formulas of forms other than $\mathbf{A}\phi$ and $\mathbf{E}\phi$, we define τ_1 as in Definition 66. As this new τ_1 does not add any atoms except for f we do not need most of τ_p and so τ becomes:

$$\tau(\phi) = \tau_1(\phi) \wedge \kappa_f \ ,$$

where κ_f is defined the same as in Section 5.1.

Translating a CTL* model M to Pair-RoCTL is now trivial, as we only have to add the $f/\mathbf{v}$ atoms to the valuation of each world of M and then add a success world s at which f and $\mathbf{v}$ are false. Given a Pair-RoCTL model to translate into a CTL* model, we either reject the model as inconsistent if it does not satisfy κ_f, or remove all worlds where f is false. As the translation of the structure does not depend on the formula being translated, this gives us an expressivity result.

Theorem 78. *For any CTL-structure M and fullpath σ through M we have $M^{\mathfrak{R}}, \sigma \vDash \tau(\phi) \iff M, \sigma \vDash \phi$.*

The proof of correctness of this new translation is similar to the previous translation and has been omitted.

6. State-RoCTL

In this section we will show that the decision problems for State-RoCTL are of similar complexity to the corresponding decision problems for CTL. We will do this by presenting a translation of State-RoCTL into CTL. Note that as we are now translating formulas into CTL we do not have to avoid using the $\mathbf{v}$ atom as CTL (and CTL*, RoCTL$^*_{\mathbf{v}}$) formulas can contain this atom. Here we find it convenient to define $\mathbf{E}\bigcirc$ and $\mathbf{P}\bigcirc$ as base operators, as this notation allows a simpler presentation of the proofs relating to State-RoCTL. Recall the formal definition of State-RoCTL given in Definition 11.

6.1. Expressivity

We will now define a translation from State-RoCTL to CTL. To understand how the translation to CTL works, consider the formula $\triangle(\alpha \mathcal{U} \beta)$. If $M, \sigma \vDash \triangle(\alpha \mathcal{U} \beta)$ then either $M, \sigma \vDash \alpha \mathcal{U} \beta$ or there exists a deviation from σ like π in Figure 1, for some integers i and n. The case where $M, \sigma \vDash \alpha \mathcal{U} \beta$ is clearly trivially represented in CTL, when translation of α and β into CTL are known.

Consider the case where $M, \sigma \nvDash \alpha U \beta$. In this case, β does not occur on π_j for $j \leq i$ as then σ would also satisfy $\alpha \mathcal{U} \beta$. And so (as in Figure 1) we see that $M, \pi_{\geq i+1} \vDash \alpha \mathcal{U} \beta$. Since π is an i-deviation $\pi_{\geq i+1}$ is failure-free and so $\pi_{i+1} \vDash \mathbf{P}(\alpha \mathcal{U} \beta)$, thus the state formula $\mathbf{E}\bigcirc\mathbf{P}(\alpha \mathcal{U} \beta)$ holds at σ_i. Thus along the path σ the state-formula α holds until $\mathbf{E}\bigcirc\mathbf{P}(\alpha \mathcal{U} \beta)$ holds together with α. The translation will recursively push $\triangle$ operators inside the $\mathcal{U}$ operator by replacing $\triangle(\alpha \mathcal{U} \beta)$ with

$$\alpha \mathcal{U} (\beta \vee (\alpha \wedge \mathbf{E}\bigcirc\mathbf{P}(\alpha \mathcal{U} \beta))) \ .$$

The $\blacktriangle$ operator is the dual of the $\triangle$, so it will be handled similarly. We now formally define the translation.

Definition 79. We now define a translation function τ from State-RoCTL formulas and path-formulas to CTL-formulas and path-formulas respectively. For any atom p we let $\tau(p) = p$. For any State-RoCTL formula α we define τ such that the following equalities hold:

$$\begin{aligned}\tau(\mathbf{E}\bigcirc\alpha) &= \mathbf{E}\bigcirc\tau(\alpha)\\ \tau(\mathbf{P}\bigcirc\alpha) &= \mathbf{E}\bigcirc(\tau(\alpha)\wedge\neg\mathbf{v})\\ \tau(\neg\alpha) &= \neg\tau(\alpha)\ .\end{aligned}$$

Likewise, for any pair α, β of State-RoCTL formulas:

$$\begin{aligned}\tau(\alpha\wedge\beta) &= \tau(\alpha)\wedge\tau(\beta)\\ \tau(\alpha\mathcal{U}\beta) &= \tau(\alpha)\,\mathcal{U}\tau(\beta)\\ \tau(\blacktriangle(\alpha\mathcal{U}\beta)) &= \tau(\alpha\wedge\mathbf{A}\bigcirc\mathbf{O}(\alpha\mathcal{U}\beta))\,\mathcal{U}\tau(\beta)\\ \tau(\triangle(\alpha\mathcal{U}\beta)) &= \tau(\alpha)\,\mathcal{U}\tau(\beta\vee(\alpha\wedge\mathbf{E}\bigcirc\mathbf{P}(\alpha\mathcal{U}\beta)))\ .\end{aligned}$$

For any State-RoCTL path-formula θ:

$$\begin{aligned}\tau(\mathbf{A}\theta) &= \mathbf{A}\tau(\theta)\\ \tau(\mathbf{E}\theta) &= \mathbf{E}\tau(\theta)\ .\end{aligned}$$

The operators $\mathbf{O}$ and $\mathbf{P}$ are not in CTL so they will have to be handled differently to $\mathbf{A}$ and $\mathbf{E}$. First we will consider the case where $\alpha\mathcal{U}\beta$ is nested directly inside the $\mathbf{O}$ or $\mathbf{P}$ operators:

$$\begin{aligned}\tau(\mathbf{O}(\alpha\mathcal{U}\beta)) &= \tau(\beta)\vee(\tau(\alpha)\wedge\mathbf{A}\bigcirc\mathbf{A}(\tau(\alpha)\,\mathcal{U}(\tau(\beta)\vee\mathbf{v})))\\ \tau(\mathbf{P}(\alpha\mathcal{U}\beta)) &= \tau(\beta)\vee(\tau(\alpha)\wedge\mathbf{E}\bigcirc\mathbf{E}((\tau(\alpha)\wedge\neg\mathbf{v})\,\mathcal{U}(\tau(\beta)\wedge\neg\mathbf{v})))\ .\end{aligned}$$

We handle the case where $\blacktriangle$ or $\triangle$ occur inside $\mathbf{O}$ or $\mathbf{P}$ by specifying that for all path-formulas θ not of the form $\alpha\mathcal{U}\beta$:

$$\begin{aligned}\tau(\mathbf{O}\theta) &= \tau(\mathbf{O}\tau(\theta))\\ \tau(\mathbf{P}\theta) &= \tau(\mathbf{P}\tau(\theta))\ ,\end{aligned}$$

and similarly for $\blacktriangle$ and $\triangle$:

$$\begin{aligned}\tau(\blacktriangle\theta) &= \tau(\blacktriangle\tau(\theta))\\ \tau(\triangle\theta) &= \tau(\triangle\tau(\theta))\ .\end{aligned}$$

We see above that $\tau(\theta)$ is of the form $\alpha\mathcal{U}\beta$ and so its definition does not require infinite recursion. For example, $\tau(\blacktriangle\blacktriangle\blacktriangle(p\mathcal{U}q)) = \tau(\blacktriangle\tau(\blacktriangle\tau(\blacktriangle(p\mathcal{U}q))))$, and we see that we can expand this from the inside out using the definition of $\tau(\blacktriangle(\alpha\mathcal{U}\beta))$ above.

To help verify that τ is well-defined (not circular), we will define a partial ordering $<$ which is to be read as "is simpler than". We will then use this to show that the recursive definition of τ is not circular as at each step of the recursion we consider simpler formulas.

Definition 80. We define a partial ordering $<$ on RoCTL* formulas such that $\phi < \psi$ if ϕ has less $\blacktriangle$ operators than ψ, and $\phi < \psi$ if ϕ and ψ have the same number of $\blacktriangle$ operators and $|\phi| < |\psi|$.

Note that we treat $\triangle$ as an abbreviation for $\neg\blacktriangle\neg$, so $p < \triangle p$. We now show that τ is not circular, and well-defined.

Lemma 81. *The formula $\tau(\phi)$ is defined for all ϕ in the domain of τ.*

Proof. We see that for all ϕ in the domain, that is all ϕ that are either State-RoCTL formulas or path-formulas, $\tau(\phi)$ occurs in the Left-Hand Side (LHS) of the above definition. We also see that where $\tau(\phi)$ occurs on the LHS and $\tau(\psi)$ occurs on the RHS, $\psi < \phi$. Thus by induction on the number of $\blacktriangle$ and induction on the length of the formulas we see that $\tau(\phi)$ is finite in length for all ϕ in the domain. □

Recall that $\tau(\phi)$ is a CTL formula. The following lemma demonstrates that State-RoCTL is expressively equivalent to CTL. The proof follows naturally from the definition of τ, but is not of trivial length as many cases need to be considered.

Lemma 82. *For all RoCTL$_\mathbf{v}$ structures M, fullpaths σ through M and State-RoCTL formulas ϕ, we have $M, \sigma \vDash \phi \iff M, \sigma \vDash \tau(\phi)$.*

Proof. For contradiction, say that ϕ is the simplest formula that provides a counter example to this lemma, that is there exists no counter example ψ such that $\psi < \phi$. In the proof of this lemma, we will use the notation $\pi_{\leq i} \cdot \sigma$ to represent the concatenation $\pi_0, \pi_1, \ldots, \pi_i, \sigma_0, \sigma_1, \ldots$ of $\pi_{\leq i}$ and σ.

We now show that ϕ does not provide a counter example for any RoCTL$_\mathbf{v}$ structure M and fullpath σ through M.

Case 1. Say that ϕ is of the form $\blacktriangle(\alpha \mathcal{U} \beta)$.

($\Longrightarrow$) Say that $M, \sigma \vDash \blacktriangle(\alpha \mathcal{U} \beta)$ and $M, \sigma \nvDash \tau(\blacktriangle(\alpha \mathcal{U} \beta))$. Thus

$$M, \sigma \nvDash \tau(\alpha \wedge \mathbf{A}\bigcirc\mathbf{O}(\alpha \mathcal{U} \beta)) \, \mathcal{U} \tau(\beta) \ ,$$

Since ϕ is the simplest counter example and $\alpha \wedge \mathbf{A}\bigcirc\mathbf{O}(\alpha \mathcal{U} \beta), \beta < \phi$ we get:

$$M, \sigma \nvDash (\alpha \wedge \mathbf{A}\bigcirc\mathbf{O}(\alpha \mathcal{U} \beta)) \, \mathcal{U} \beta \ .$$

There are two ways to avoid σ satisfying the above formula. The first is if for all i, we have $M, \sigma \nvDash \beta$; however, $M, \sigma \vDash \blacktriangle(\alpha \mathcal{U} \beta)$ so we know that $M, \sigma \vDash \alpha \mathcal{U} \beta$ and there must exist some i such that $M, \sigma_i \vDash \beta$. The second way is if there exists an integer i such that for all $j \leq i$ we have $M, \sigma_j \nvDash \beta$, and

$$M, \sigma_i \nvDash (\alpha \wedge \mathbf{A}\bigcirc\mathbf{O}(\alpha \mathcal{U} \beta)) \ .$$

Assume without loss of generality that j is the smallest j such that $M, \sigma_j \nvDash \beta$. Since $M, \sigma \vDash \alpha \mathcal{U} \beta$ and $M, \sigma_i \nvDash \beta$ we see that $M, \sigma_i \vDash \alpha$ and hence

$$M, \sigma_i \nvDash \mathbf{A}\bigcirc\mathbf{O}(\alpha \mathcal{U} \beta) \ .$$

The sequence of operators $\mathbf{A}\bigcirc\mathbf{O}$ quantifies over exactly those fullpaths that are failure-free after the first step, or in other words 0-deviations. Thus there exists an i-deviation π from σ such that $M, \pi_{\geq i} \not\models \bigcirc\alpha\mathcal{U}\beta$ and equivalently $M, \pi_{\geq i+1} \not\models \alpha\mathcal{U}\beta$. Recall that $M, \sigma_j \not\models \beta$ for all $j \leq i$; since β is a state-formula and $\sigma_{\leq i} = \pi_{\leq i}$ it follows that $M, \pi_j \not\models \beta$. Combining this with the fact that $M, \pi_{\geq i+1} \not\models \alpha\mathcal{U}\beta$ we find that $M, \pi \not\models \alpha\mathcal{U}\beta$. Since π is a deviation from σ we find that $M, \sigma \not\models \blacktriangle(\alpha\mathcal{U}\beta)$, which contradicts our original assumption.

We now consider the reverse part of the case where ϕ is of the form $\blacktriangle(\alpha\mathcal{U}\beta)$.

($\Longleftarrow$) Say that $M, \sigma \not\models \blacktriangle(\alpha\mathcal{U}\beta)$ and $M, \sigma \models \tau(\blacktriangle(\alpha\mathcal{U}\beta))$. Since $M, \sigma \not\models \blacktriangle(\alpha\mathcal{U}\beta)$ either $M, \sigma \not\models \alpha\mathcal{U}\beta$ or there exists a deviation π from σ such that $M, \pi \not\models \alpha\mathcal{U}\beta$. If $M, \sigma \not\models \alpha\mathcal{U}\beta$ then clearly $M, \sigma \not\models (\alpha \wedge \mathbf{A}\bigcirc\mathbf{O}(\alpha\mathcal{U}\beta))\,\mathcal{U}\beta$, so by contradiction there must exist an i-deviation π from σ such that $M, \pi \not\models \alpha\mathcal{U}\beta$.

Since $M, \sigma \models (\alpha \wedge \mathbf{A}\bigcirc\mathbf{O}(\alpha\mathcal{U}\beta))\,\mathcal{U}\beta$ we see that there exists n such that $M, \sigma_n \models \beta$ and for all $m < n$ it is the case that

$$M, \sigma_m \models (\alpha \wedge \mathbf{A}\bigcirc\mathbf{O}(\alpha\mathcal{U}\beta)) \ .$$

Say that $n \leq i$. Since $M, \sigma_m \models \alpha$ for all $m < n$ and $M, \sigma_n \models \beta$, we can see that as $\pi_{\leq i} = \sigma_{\leq i}$ it must also be the case that $M, \pi \models (\alpha\mathcal{U}\beta)$, as in Figure 2.

However, recall that we chose the path π such that $M, \pi \not\models \alpha\mathcal{U}\beta$. By contradiction we know that $n > i$.

Since $n > i$ we know that for all $j \leq i$ it is the case that $M, \sigma_j \models \alpha$ and $\sigma_j = \pi_j$. From this and the fact that $M, \pi \not\models \alpha\mathcal{U}\beta$ it follows that $M, \pi_{\geq i+1} \not\models \alpha\mathcal{U}\beta$. Since $\pi_{\geq i+1}$ is failure-free we see that $M, \pi_{i+1} \not\models \mathbf{O}(\alpha\mathcal{U}\beta)$ and

$$M, \pi_i \not\models \mathbf{A}\bigcirc\mathbf{O}(\alpha\mathcal{U}\beta) \ .$$

Since $\pi_i = \sigma_i$ we also have $M, \sigma_i \not\models \mathbf{A}\bigcirc\mathbf{O}(\alpha\mathcal{U}\beta)$. However, recall that

$$M, \sigma_m \models (\alpha \wedge \mathbf{A}\bigcirc\mathbf{O}(\alpha\mathcal{U}\beta)) \ ,$$

for all $m < n$. By contradiction we see that the smallest counter-example ϕ cannot be of the form $\blacktriangle(\alpha\mathcal{U}\beta)$. The proof for the case where ϕ is of the form $\triangle(\alpha\mathcal{U}\beta)$ is similar.

Case 2. Say that ϕ is of the form $\triangle(\alpha\mathcal{U}\beta)$ then recall that

$$\tau(\triangle(\alpha\mathcal{U}\beta)) = \tau(\alpha)\,\mathcal{U}\tau(\beta \vee (\alpha \wedge \mathbf{E}\bigcirc\mathbf{P}(\alpha\mathcal{U}\beta))) \ .$$

Say that $M, \sigma \models \triangle(\alpha\mathcal{U}\beta)$ and $M, \sigma \not\models \tau(\triangle(\alpha\mathcal{U}\beta))$. Thus

$$M, \sigma \not\models \alpha\mathcal{U}(\beta \vee (\alpha \wedge \mathbf{E}\bigcirc\mathbf{P}(\alpha\mathcal{U}\beta))) \ ,$$

and clearly $M, \sigma \not\models \alpha\mathcal{U}\beta$. Hence there exists an i-deviation π from σ such that $M, \pi \models \alpha\mathcal{U}\beta$, for some $i \in \mathbb{N}$. Since $M, \sigma \not\models \alpha\mathcal{U}\beta$ we see that $M, \sigma_j \not\models \beta$ for any $j \leq i$. Thus $M, \pi_{\geq i+1} \models \alpha\mathcal{U}\beta$, and since $M, \pi_{\geq i+1}$ is failure-free we know that $M, \pi_{i+1} \models \mathbf{P}(\alpha\mathcal{U}\beta)$. From this we know that $M, \sigma_i \models \mathbf{E}\bigcirc\mathbf{P}(\alpha\mathcal{U}\beta)$. Since $M, \pi \models \alpha\mathcal{U}\beta$ and β is not satisfied before π deviates from σ we know that for each $j \leq i$ we have $M, \sigma_j \models \alpha$. Hence $M, \sigma \models (\alpha\mathcal{U}(\alpha \wedge \mathbf{E}\bigcirc\mathbf{P}(\alpha\mathcal{U}\beta)))$ and so $M, \sigma \models \tau(\triangle(\alpha\mathcal{U}\beta))$.

Case 3. Say that ϕ is of the form $\mathbf{P}\bigcirc\alpha$.

($\Longrightarrow$) Say that $M, \sigma \vDash \mathbf{P}\bigcirc\alpha$. Since $M, \sigma \vDash \mathbf{P}\bigcirc\alpha$ we see there exists a failure-free fullpath π such that $\pi_0 = \sigma_0$ and $M, \pi \vDash \bigcirc\alpha$ so $M, \pi_{\geq 1} \vDash \alpha$. Since π is failure-free we see that $M, \pi_{\geq 1} \vDash \neg\mathbf{v}$, and hence $M, \pi_{\geq 1} \vDash \alpha \wedge \neg\mathbf{v}$. As $M, \pi \vDash \bigcirc(\alpha \wedge \neg\mathbf{v})$ and $\pi_0 = \sigma_0$ we see that $M, \pi \vDash \mathbf{E}\bigcirc(\alpha \wedge \neg\mathbf{v})$. This is precisely $\tau(\mathbf{P}\bigcirc\alpha)$.

($\Longleftarrow$) Say that $M, \sigma \vDash \tau(\mathbf{P}\bigcirc\alpha) = \mathbf{E}\bigcirc(\alpha \wedge \neg\mathbf{v})$. Then there exists π such that $\pi_0 = \sigma_0$ and $M, \pi \vDash \bigcirc(\alpha \wedge \neg\mathbf{v})$. Thus $M, \pi_{\geq 1} \vDash (\alpha \wedge \neg\mathbf{v})$. We can construct a failure-free fullpath ρ starting at π_1, We note that as α is a state formula and $M, \pi_{\geq 1} \vDash \alpha$ the fullpath ρ also satisfies α. Since ρ is failure-free and satisfies $\neg\mathbf{v}$, we see that $\sigma_0 \cdot \rho$ is failure-free. Note that $M, \sigma_0 \cdot \rho \vDash \bigcirc\alpha$. It finally follows that $M, \sigma \vDash \mathbf{P}\bigcirc\alpha$.

Case 4. Say that ϕ is of the form $\mathbf{O}(\alpha\mathcal{U}\beta)$.

($\Longrightarrow$) Say that $M, \sigma \vDash \mathbf{O}(\alpha\mathcal{U}\beta)$. If $M, \sigma \vDash \beta$ then, as ϕ is the simplest counter-example and β is simpler, $M, \sigma \vDash \tau(\beta)$ and $M, \sigma \vDash \tau(\mathbf{O}(\alpha\mathcal{U}\beta))$.

If $M, \sigma \nvDash \beta$ then clearly $M, \sigma \vDash \alpha$ and $M, \sigma \vDash \tau(\alpha)$. For contradiction, consider a fullpath π such that $\pi_0 = \sigma_0$ and $M, \pi_{\geq 1} \nvDash \alpha\mathcal{U}(\beta \vee \mathbf{v})$; we see that there exists $i \in \mathbb{N}$ such that $M, \pi_{\geq j} \nvDash \tau(\beta) \vee \mathbf{v}$ for all $j \leq i$ and $M, \pi_{\geq i} \nvDash \tau(\alpha) \vee \tau(\beta) \vee \mathbf{v}$. We see that we can construct a failure-free path ρ such that $\rho_0 = \pi_i$. We see that $\pi_{\leq i-1} \cdot \rho$ is failure-free, and that $M, \pi_{\leq i-1} \cdot \rho \nvDash \alpha\mathcal{U}\beta$. This implies that $M, \sigma \nvDash \mathbf{O}(\alpha\mathcal{U}\beta)$, and so by contradiction we know that $M, \pi_{\geq 1} \vDash \tau(\alpha)\,\mathcal{U}\,(\tau(\beta) \vee \mathbf{v})$ for all paths π starting at σ_0. It is then easy to show that $M, \sigma \vDash \mathbf{A}\bigcirc\mathbf{A}(\alpha\mathcal{U}(\beta \vee \mathbf{v}))$ and since $M, \sigma \vDash \alpha$ we see that $M, \sigma \vDash \tau(\mathbf{O}(\alpha\mathcal{U}\beta))$.

($\Longleftarrow$) Say that $M, \sigma \nvDash \mathbf{O}(\alpha\mathcal{U}\beta)$ and $M, \sigma \vDash \tau(\mathbf{O}(\alpha\mathcal{U}\beta))$. We see that if $M, \sigma \vDash \tau(\beta)$ then $M, \sigma \vDash \beta$ and since β is a state formula, $M, \sigma \vDash \mathbf{O}(\alpha\mathcal{U}\beta)$. Say that instead

$$M, \sigma \vDash \tau(\alpha) \wedge \mathbf{A}\bigcirc\mathbf{A}(\tau(\alpha)\,\mathcal{U}\,(\tau(\beta) \vee \mathbf{v}))\ .$$

Then we see that for every path π starting at σ_0 we have

$$M, \pi \vDash \tau(\alpha) \wedge \bigcirc(\tau(\alpha)\,\mathcal{U}\,(\tau(\beta) \vee \mathbf{v}))\ ,$$

and $\alpha \wedge \bigcirc(\alpha\mathcal{U}(\beta \vee \mathbf{v}))$. Let π be failure-free, then we have $M, \pi \vDash \bigcirc\Box\neg\mathbf{v}$ and so $M, \pi \vDash \alpha \wedge \bigcirc(\alpha\mathcal{U}\beta)$. We see that $M, \pi \vDash \alpha\mathcal{U}\beta$ for all failure-free full paths π starting at σ_0 and so $M, \sigma \vDash \mathbf{O}(\alpha\mathcal{U}\beta)$.

Case 5. The case where ϕ is of the form $\mathbf{P}(\alpha\mathcal{U}\beta)$ is similar to $\mathbf{O}(\alpha\mathcal{U}\beta)$.

Say that ϕ is of the form $\neg\alpha$. By definition $\tau(\neg\alpha) = \neg\tau(\alpha)$. State-RoCTL is a syntactic restriction of RoCTL*, so since τ leaves the $\neg$ unchanged we see that

$$M, \sigma \vDash \neg\alpha \iff M, \sigma \vDash \tau(\neg\alpha)\ .$$

Where ϕ is of the form $\mathbf{E}\bigcirc\alpha$, $\alpha \wedge \beta$, $\alpha\mathcal{U}\beta$, $\mathbf{O}\theta$, $\mathbf{P}\theta$, $\blacktriangle\theta$, $\triangle\theta$, $\mathbf{A}\theta$, or $\mathbf{E}\theta$, we likewise see that, as State-RoCTL is a syntactic restriction, we have $M, \sigma \vDash \phi \iff M, \sigma \vDash \tau(\phi)$. $\square$

6.2. Complexity

We have shown that every statement in State-RoCTL can be expressed in CTL when the CTL logic formulas are allowed to reference the special atom **v**. The translation into CTL isn't linear. For example, consider

$$\tau\left(\triangle\left(\alpha \mathcal{U} \beta\right)\right) = \tau\left(\alpha\right) \mathcal{U} \tau\left(\beta \vee \left(\alpha \wedge \mathbf{E}\bigcirc\mathbf{P}\left(\alpha \mathcal{U} \beta\right)\right)\right) \ .$$

See that α and β occur twice on the Right Hand Sided (RHS). Thus the length of the translated formula can double with each $\triangle$. However, since α and β are state formulas, it is well known by implementers of decision procedures that we can replace α and β with atoms p_α and p_β. Elsewhere a clause is added requiring that

$$\mathbf{A}\Box\left(p_\alpha \leftrightarrow \tau\left(\alpha\right) \wedge p_\beta \leftrightarrow \tau\left(\beta\right)\right) \ .$$

A similar trick can be used with model checkers, where the atom p_α is added to states where the model checker determines that α holds. We will discuss the details of how this can be achieved for State-RoCTL. Since we are translating the State-RoCTL formula into CTL*, we will explicitly state this property for CTL* as a lemma.

Lemma 83. *Say that in some CTL-structure M there exists an atom p_ψ such that for all fullpaths σ we have $M, \sigma \vDash p_\psi$ iff $M, \sigma \vDash \psi$. Then for any CTL* formula ϕ we have $M, \sigma \vDash \phi[\psi/p_\psi] \iff M, \sigma \vDash \phi$.*

It is trivial to prove this lemma inductively from the semantics of CTL*. While the lemma above holds even if ψ is not a state-formula, we will only apply it to state formulas, as in the next corollary.

Corollary 84. *For any CTL* formula ϕ, and state-formula ψ, it is the case that ϕ is satisfiable iff $\hat{\phi}$ is satisfiable where $\hat{\phi} = \phi[\psi/p_\psi] \wedge \mathbf{A}\Box\left(p_\psi \leftrightarrow \psi\right)$.*

Proof. Say that $\hat{\phi}$ is satisfied on some structure M and fullpath π. We can say WLOG that every world in the structure M is accessible from π_0. Since $M, \sigma \vDash \mathbf{A}\Box\left(p_\psi \leftrightarrow \psi\right)$, we know that $M, \sigma \vDash p_\psi \iff M, \sigma \vDash \psi$ for all fullpaths σ through M. Thus from the previous lemma and the fact that $M, \pi \vDash \phi[\psi/p_\psi]$ we know that $M, \pi \vDash \phi$. Hence ϕ is satisfiable.

Say that ϕ is satisfied on some structure M and fullpath σ. Since ψ is a state-formula we can add p_ψ to M to produce M' such that p_ψ is true on exactly those states where ψ is true. Hence $M', \sigma \vDash \mathbf{A}\Box\left(p_\psi \leftrightarrow \psi\right)$ and from the previous lemma $M', \sigma \vDash \mathbf{A}\Box\left(p_\psi \leftrightarrow \psi\right)$. □

We now show that, as expected, avoiding duplication of subformulas results in an easily computable translation of linear size.

Theorem 85. *There exists a polynomial-time computable satisfiability preserving linear translation $\tau_{\boldsymbol{sat}}$ from State-RoCTL formulas into CTL formulas.*

Proof. Recall that

$$\tau\left(\blacktriangle\left(\alpha\mathcal{U}\beta\right)\right)=\tau\left(\alpha\wedge\mathbf{A}\bigcirc\mathbf{O}\left(\alpha\mathcal{U}\beta\right)\right)\mathcal{U}\tau\left(\beta\right)$$
$$\tau\left(\triangle\left(\alpha\mathcal{U}\beta\right)\right)=\tau\left(\alpha\right)\mathcal{U}\tau\left(\beta\vee\left(\alpha\wedge\mathbf{E}\bigcirc\mathbf{P}\left(\alpha\mathcal{U}\beta\right)\right)\right)\ .$$

We define a translation function τ' similarly to τ with the exception that

$$\tau'\left(\blacktriangle\left(\alpha\mathcal{U}\beta\right)\right)=\tau'\left(p_{\tau'(\alpha)}\wedge\mathbf{A}\bigcirc\mathbf{O}\left(p_{\tau'(\alpha)}\mathcal{U}p_{\tau'(\beta)}\right)\right)\mathcal{U}p_{\tau'(\beta)}$$
$$\tau'\left(\triangle\left(\alpha\mathcal{U}\beta\right)\right)=\tau'\left(p_{\tau'(\alpha)}\right)\mathcal{U}\tau\left(p_{\tau'(\beta)}\vee\left(p_{\tau'(\alpha)}\wedge\mathbf{E}\bigcirc\mathbf{P}\left(p_{\tau'(\alpha)}\mathcal{U}p_{\tau'(\beta)}\right)\right)\right)\ .$$

Below, recall that $\sqsubseteq$ is read as "subformula of". We then define a translation function

$$\tau''\left(\alpha\right)=\tau'\left(\alpha\right)\wedge\bigwedge_{\beta\text{ s.t. }p_\beta\sqsubseteq(\tau'(\alpha))}\mathbf{A}\Box\left(p_\beta\leftrightarrow\beta\right)\ .$$

Recall that we need the clause $\mathbf{A}\Box\mathbf{E}\bigcirc\neg\mathbf{v}$ to ensure that the translated formula is only satisfied on RoCTL$_\mathbf{v}$ structure, so we let

$$\tau_{\mathbf{sat}}=\tau''\left(\alpha\right)\wedge\mathbf{A}\Box\mathbf{E}\bigcirc\neg\mathbf{v}\ .$$

From the previous corollary we see that $\tau''\left(\alpha\right)$ is satisfiable iff $\tau\left(\alpha\right)$ (and α) is satisfiable. We will now show that for any State-RoCTL* formula α it is the case that $\left|\tau''\left(\alpha\right)\right|\leq 45\left|\alpha\right|$, and hence that the translation $\tau''\left(\alpha\right)$ is linear. Other authors often use slightly different set of primitive operators, such as using $\vee$ as a primitive operator instead of $\wedge$. In such variations it may not be the case that $\left|\tau''\left(\alpha\right)\right|\leq 45\left|\alpha\right|$. Nevertheless translating between such minor syntactic changes is linear so, the linearity of the translation τ'' is preserved even if the precise factor 45 is not.

For compatibility with the definition of the length of formulas, in this proof we will not include (and) in the count of symbols in a formula. We can enumerate the symbols of α, and modify $\tau^{'}$ such that $\tau^{'}$ preserves this enumeration. Below we give an example of assigning an integer label i to the operators on the RHS from a label on the LHS. These labels do not have any semantic meaning, they are only added to assist in counting the number of operators used. Note also that we have expanded the $\vee$ operator into two the base operators $\neg$ and $\wedge$. We define $\tau'\left(\triangle^i\left(\alpha\mathcal{U}^j\beta\right)\right)$ to be equal to:

$$\tau'\left(p^i_{\tau'(\alpha)}\right)\mathcal{U}^i\tau\left(\neg^i\left(\neg^i p^i_{\tau'(\beta)}\wedge^i\neg^i\left(p^i_{\tau'(\alpha)}\wedge^i\mathbf{E}^i\bigcirc^i\mathbf{P}^i\left(p^i_{\tau'(\alpha)}\mathcal{U}^i p^i_{\tau'(\beta)}\right)\right)\right)\right)\ .$$

Likewise for the other lines defining τ' we define τ' such that the new operators introduced on the RHS have the same label as the operator of the highest precedence on the LHS. We see that in the line above, there are 15 symbols labelled with i for the one operator $\triangle$ labelled with i on the LHS. It is easy to see from induction that i occurs at most 15 times in $\tau''\left(\alpha\right)$ for each time it occurs in α. Note that τ'' contains $\leftrightarrow$ operators, which we define as abbreviations

rather than as primitive operators. Since these are not nested, expanding each occurrence of $\cdots \wedge \mathbf{A}\Box\left(p_\beta \leftrightarrow \beta\right)$ into

$$\cdots \wedge \mathbf{A}\Box\left(\neg\left(\neg\left(p_\beta \wedge \beta\right) \wedge \neg\left(\neg p_\beta \wedge \neg\beta\right)\right)\right) \ ,$$

at most doubles the number of i-labelled symbols. Hence for each i-labelled symbol in α there are at most 30 i-labelled symbols in $\tau''\left(\alpha\right)$. We see in the equation fragment above (excluding the portion represented as $\cdots$) has 15 unlabelled symbols. Thus for each symbol in α there are at most 45 symbols in $\tau''\left(\alpha\right)$. Thus $\left|\tau''\left(\alpha\right)\right| \leq 45\left|\alpha\right|$. □

We see that the translation function $\tau_{\mathbf{sat}}$ is itself computationally simple. Since the decision problem for CTL is in EXPTIME [20], we get the following theorem.

Theorem 86. *The satisfiability decision problem for State-RoCTL is in EXPTIME.*

We will now show that, as with CTL [24], we can model check State-RoCTL in $\mathcal{O}\left(\left(\left|S\right| + \left|R\right|\right) \cdot \left|\psi\right|\right)$ time.

Theorem 87. *Given a State-RoCTL structure $M = (S, R, g)$, and formula ψ the State-RoCTL model checking problem can be decided in time of order $\mathcal{O}\left(\left(\left|S\right| + \left|R\right|\right) \cdot \left|\psi\right|\right)$.*

Proof. It is easy to see that model-checking procedure is polynomial; for each atom $p_\beta \sqsubseteq \tau''\left(\psi\right)$ and world $w \in S$ we can use the CTL model checking procedure to model check β at the world w and add p_β recursively to the model if $M, w \vDash \beta$. We then see that

$$M, w \vDash \tau'\left(\psi\right) \iff M, w \vDash \tau'\left(\psi\right) \iff M, w \vDash \psi \ .$$

Hence model checking the State-RoCTL formula ψ will result in at most $\left|S\right| \cdot \left|\psi\right|$ calls to the $\mathcal{O}\left(\left(\left|S\right| + \left|R\right|\right) \cdot \left|\psi\right|\right)$ model checking procedure. We will now refine this procedure.

The model checking procedure for CTL in [24] marks each state with the subformulas of β that hold at each state when checking whether $M, w \vDash \beta$ holds for some formula β, structure M and world w. This is done so that each formula of length n can be easily model checked once each state has been marked with the subformulas of length $n - 1$. After performing the CTL model checking procedure of [24] we can inspect this marking to determine at which worlds of M the formula β holds at, and we do not need to perform the procedure for each world in S. As such we need to call the CTL model-checking procedure of [24] at most $\left|\psi\right|$ times. This refinement has removed a factor of $\left|S\right|$. We will now show that algorithm achieves a complexity of $\mathcal{O}\left(\left(\left|S\right| + \left|R\right|\right) \cdot \left|\psi\right|\right)$ despite needing to call the CTL model-checker multiple times. The basis of this proof is the fact that we only need to model check short formulas, which are in total shorter than $\tau''\left(\psi\right)$.

Let Φ be the set of formulas that we send to the CTL model-checker, that is

$$\Phi = \{\tau'(\psi)\} \cup \{\beta : p_\beta \sqsubseteq \tau'(\psi)\} \ .$$

Hence the time required is

$$\mathcal{O}\left(\sum_{\phi\in\Phi}((|S|+|R|)\cdot|\phi|)\right) ,$$

which simplifies to

$$\mathcal{O}\left((|S|+|R|)\cdot\sum_{\phi\in\Phi}|\phi|\right) .$$

Note that each member of Φ is a subformula of $\tau''(\psi)$. Since each member of Φ comes from a separate part of $\tau''(\psi)$ we see that $\sum_{\phi\in\Phi}|\phi| \leq |\tau''(\psi)| \leq 45\,|\psi|$. As such the complexity is $\mathcal{O}((|S|+|R|)\cdot|\psi|)$. □

It is well known that model-checking CTL formulas is P-hard [21]. As State-RoCTL is an extension of CTL it is clear that it is also P-hard. Together with the previous theorem, this gives us the following corollary.

Corollary 88. *Model checking State-RoCTL formulas is P-complete.*

7. Related Work

In this paper we have discussed sublogics of the logic of robustness RoCTL*. Many previous logics for reasoning about fault tolerant systems represent reliability using probabilities. For examples of such logics, see [40, 41, 42, 43]. With such logics, if we know the probabilities of various forms of failure, we can determine the probability that the system will continue to function correctly.

Another approach is to find the smallest metric perturbation that is required to cause failure. This can be useful when processing noisy signals [44, 45], where the signal is generally analog at the hardware level, and in biological systems [46]. A logic has been proposed to model how robustly a system can handle non-compliance by some set of agents [47, 48]. We will now discuss these two basic approaches in greater detail.

The approach of [44, 45] is to define the robustness of a Metric Temporal Logic (MTL) formula. MTL extends the syntax of LTL by replacing the $\mathcal{U}$ operator with a $\mathcal{U}_\mathcal{I}$ operator where $\mathcal{I}$ represents an (possibly continuous) interval. The statement $\phi\mathcal{U}_\mathcal{I}\psi$ indicates that within the interval $\mathcal{I}$ it is the case that ϕ is true until ψ becomes true. They define how robustly true a statement is in terms of the closest signal which does not satisfy the formula. The semantics of their logic considers not only whether an atom p is true or false, but also the set of signals "$O(p)$" on which p is true. They define a measure of distance the $\mathbf{Dist}_d(x,S)$ of a point x to a set S on a metric d such that

the distance is negative if the point is inside the set and positive if the point is outside. They define the "robust valuation" of an atomic proposition p as $[[p, O]]C(s,t) := \mathbf{Dist}_d(s(t), O(p))$, where O is an "observation map" [45, p22], s is the signal, t is the time, $s\,(t)$ is the current value of the signal, and $O(p)$ is the set of values where p is true. They then define the robust valuation of other formulas recursively in terms of the atomic proposition.

The approach of [46] is broadly similar to that of [44, 45], but has a couple of differences. Like [44, 45], they define robustness in terms of metric perturbations. Unlike [44, 45], [46] considers LTL rather than MTL formulas. Additionally, they do not redefine the semantics of the logic, instead they define the "violation degree" of a trace to be the distance to the closest trace that satisfies a formula.

The Probabilistic real time Computation Tree Logic (PCTL) of [40] is a variation of CTL. In place of the all paths operator $\mathbf{A}$ they use $[\phi]_{\geq p}$ and $[\phi]_{>p}$ which indicates that, from the current world, the probability of the path formula ϕ occurring is at least p and greater than p respectively. In a manner similar to MTL, they also replace the $\mathcal{U}$ operator with a $\mathcal{U}^{\leq t}$ operator. The formula $\phi\mathcal{U}^{\leq t}\psi$ indicates that ψ will become true within the next t time steps, and prior to then ϕ will remain true.

As RoCTL* has an "obligatory" operator, it can be considered as a deontic logic. Early deontic logic has been plagued with paradoxes. The first deontic logic was that of Mally. Under Mally's logic it is provable that everything is as it ought to be ([49]; cited in [50, 51]). Although Mally considered this to be an important philosophical result, it reduces the expressiveness of the logic to that of classical logic. RoCTL* is primarily intended to model robustness of systems to violations of soft constraints rather than to give deep philosophical insights into the nature of obligation. For example, $\mathbf{A}\phi \rightarrow \mathbf{O}\phi$ is valid in RoCTL*. When discussing a system with soft constraints this is most naturally interpreted as the uncontroversial statement "If ϕ is a hard constraint, it is also a soft constraint". However, from a philosophical perspective it would be interpreted as "If ϕ is inevitable, it is also obligatory", which philosophers following Mally have found questionable. Further comparison of RoCTL* with deontic logics is found in the PhD thesis [5].

When developing a normative system it is reasonable to ask whether the normative system is robust to non-compliance by some subset of the agents. Recently [47] have considered necessary and sufficient coalitions needed to comply with a normative system to ensure the goals of the normative system are achieved, and proposed a logic to reason about such issues. There has been considerable interest in this area [52, 48, 53].

Alternating Tree Logic [54] is an extension of CTL* that can also be used to reason about coalitions. This logic replaces the CTL* operator $\mathbf{A}$ with a $\langle\langle\mathcal{C}\rangle\rangle$ operator that represents "if the agents in the coalition $\mathcal{C}$ work together, they can achieve…". ATL* can be used to reason about robustness of a system to misbehaving subsystems or users. This is an important form of robustness, but is very different from transient faults.

Graded modalities have been used in a range of modal logics to extend

universal and existential quantifiers to represent "for all but k successors" and "there exists at least k successors". For example, a graded version of the μ-calculus [55] is provided in [56]. This is a propositional modal logic with least and greatest fixpoint operators and graded modalities. Graded versions of CTL have been considered in [57, 58] and for CTL* in [59]. The graded modalities for CTL/CTL* consider the number of paths rather than the number of successors (as in the graded μ-calculus). This can be used to quantify the robustness of systems by considering the numbers of paths showing good or bad properties. The notion of robustness for these graded logics is different to what we consider which is whether properties hold given a fixed number of failures, rather than the number of paths where a property holds. Different semantics have been adopted for the graded modalities depending on how different paths are defined. The satisfiability problem is considered in [57, 58, 59] and model checking in [57, 59].

A graded version of ATL (called Graded ATL) [60] has some ability to reason about transient faults. It can represent the number of paths or strategies that are available to reach some goal. This is quite different from RoCTL*. Graded ATL has a number of different semantics, "online", "memoryless", and "offline". However, none of these semantics quite capture the idea of always achieving a goal in the face of some bounded number of faults. For example, a river with several slippery fords may offer you multiple strategies to get a book to the other side, but a single misstep will still ruin the book.

Graded ATL is well suited to representing robustness in situations where we can start again from the beginning. For example, if a random computer glitch prevents a sequence of file converters from correctly processing a file, a different sequence could be tried, and an increase in the number of combinations of converters that should work increases the robustness of the system. When starting again is not an option, more strategies does not necessarily mean that our system is more robust. Graded versions [61, 62] also exist for Strategy Logics. Strategy logics are more expressive than ATL and ATL* allowing quantification over strategies.

8. Conclusion

We discussed three sub-logics of RoCTL*: RoBCTL*, Pair-RoCTL and State-RoCTL. We have given a tableau for RoBCTL*, shown that Pair-RoCTL is not any easier to decide than CTL*, and shown that existing techniques can be used to solve State-RoCTL* satisfiability and model checking problems with no increase in computational complexity. We have not given a model checking procedure for RoBCTL*. RoCTL* can be model checked by translating the formulas into CTL*, and then model checking the resulting formulas [6]. RoBCTL* can likewise be translated into BCTL*; however, whereas CTL* has the bounded model property [36], BCTL* models have bundles which are never finite. Thus the RoBCTL*/BCTL* model checking property cannot even be defined without reference to some particular finite representation BCTL* models. Although BCTL* tableaux provide such a finite representation, they may not be the most natural representation.

We presented a tableau based decision procedure for RoBCTL*, which allows us to reason about robustness in systems that need not be limit closed. The complexity of this decision procedure may be non-elementary; however, we discussed why this may not be a problem for most applications. For example, the examples in Section 3 had the property that they never had $\mathcal{U}$, $\blacktriangle$, $\mathcal{U}$ nested in order, which is sufficient to ensure that the decision procedure for RoBCTL* terminates in an elementary amount of time. In [6] it was shown that RoCTL* could be decided by reducing to QCTL* and tree automata, and further that no such reductions could provide an elementary decision procedure for RoCTL*. The tableau for RoBCTL* was first presented in a conference paper [28], and no optimisation has been found to provide an elementary decision procedure for RoBCTL* or RoCTL*. This suggests the possibility that no elementary decision procedure for RoCTL* will be found. For this reason we also investigated syntactic restrictions of RoCTL*.

We have shown that although the syntax of Pair-RoCTL is an intuitive definition of a CTL-like restriction of RoCTL*, the properties of Pair-RoCTL are closer to CTL*. We have shown that the addition of a "success world" to CTL* models allows every property that can be expressed in CTL* to be expressed in Pair-RoCTL. Combining this with the result [6] that every property that can be expressed in RoCTL* can be expressed in CTL* when CTL* formulas are allowed to include the special violation atom indicates that Pair-RoCTL, CTL* and RoCTL* all have similar expressivity. Additionally, we have shown that the decision problems of Pair-RoCTL are in complexity classes that are at least as hard as those of CTL*, for example, satisfiability checking Pair-RoCTL is 2-EXPTIME hard. Determining whether the decision problems for Pair-RoCTL are as hard as those RoCTL* remains an open problem.

State-RoCTL is as easy to reason with as CTL. State-RoCTL* has a linear length satisfiability-preserving translation into CTL that is efficient to compute. This allows existing implementations of CTL satisfiability procedures to be used to decide State-RoCTL. Even existing model checking procedures for CTL can be used on the linear translation with only trivial modifications; that is recursively adding each p_α atom in the translation at states of the model where α is satisfied. Although having CTL-like complexity, State-RoCTL can naturally express non-trivial RoCTL* formulas, such as formulas that have Prone nested directly within Robustly.

9. Acknowledgements

This project is supported by the Australian Government's International Science Linkages program and the Australian Research Council. C. Dixon was partially supported by the EPSRC funded project Trustworthy Robotic Assistants (EP/K006193/1) and EPSRC funded RAI Hub RAIN (EP/R026084/1). We would like to thank the referees for their detailed comments that have helped us improve the paper.

References

[1] T. French, J. C. M^{c}Cabe-Dansted, M. Reynolds, A complete axiomatization of a temporal logic with obligation and robustness, Journal of Logic and Computation 26 (5) (2016) 1439–1467.

[2] A. Pnueli, The Temporal Logic of Programs, in: Proceedings of the 18th Annual Symposium on Foundations of Computer Science, IEEE, 46–57, 1977.

[3] E. A. Emerson, E. M. Clarke, Using branching time temporal logic to synthesize synchronization skeletons, Science of Computer Programming 2 (3) (1982) 241–266.

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

[5] J. C. M^{c}Cabe-Dansted, A Temporal Logic of Robustness, Ph.D. thesis, The University of Western Australia, Perth, Australia, 2011.

[6] J. C. M^{c}Cabe-Dansted, T. French, S. Pinchinat, M. Reynolds, Expressiveness and succinctness of a logic of robustness, Journal of Applied Non-Classical Logics 25 (3) (2015) 193–228.

[7] O. Kupferman, Augmenting branching temporal logics with existential quantification over atomic propositions, in: O. Grumberg (Ed.), Proceedings of the 7th International Conference on Computer Aided Verification, Springer, Liege, 325–338, 1995.

[8] O. Kupferman, Augmenting branching temporal logics with existential quantification over atomic propositions, Journal of Logic and Computation 9 (2) (1999) 135–147.

[9] M. Reynolds, A Tableau for Bundled CTL*, Journal of Logic and Computation 17 (1) (2007) 117–132.

[10] M. Reynolds, A tableau-based decision procedure for CTL*, Journal of Formal Aspects of Computing 23 (2011) 1–41.

[11] O. Friedmann, M. Latte, M. Lange, A Decision Procedure for CTL* Based on Tableaux and Automata, in: J. Giesl, R. Hähnle (Eds.), IJCAR, vol. 6173, Springer, 331–345, 2010.

[12] J. C. M^{c}Cabe-Dansted, A Rooted Tableau for BCTL*, in: International Methods for Modalities Workshop, Electronic Notes in Theoretical Computer Science, vol. 278, Elsevier, ISSN 1571-0661, 145–158, 2011.

[13] O. Friedmann, M. Lange, M. Latte, Satisfiability Games for Branching-Time Logics, Logical Methods in Computer Science [electronic only] 9.

[14] J. C. M^{c}Cabe-Dansted, M. Reynolds, Fairness with EXPTIME Bundled CTL Tableau, in: A. Cesta, C. Combi, F. Laroussinie (Eds.), Proceedings of the International Symposium on Temporal Representation and Reasoning, 164–173, 2014.

[15] J. C. M^{c}Cabe-Dansted, M. Reynolds, To be fair, use bundles, Annals of Mathematics and Artificial Intelligence 80 (3-4) (2017) 317–364.

[16] E. M. Clarke, O. Grumberg, H. Hiraishi, S. Jha, D. D. E. Long, K. L. McMillan, L. A. Ness, Verification of the Futurebus+ cache coherence protocol, Formal Methods in System Design 6 (2) (1995) 217–232, ISSN 1572-8102.

[17] Proceedings of the ASPDAC 2002/VLSI Design 2002, CD-ROM, 7-11 January 2002, Bangalore, India, IEEE Computer Society, 2002.

[18] M. Lahijanian, S. B. Andersson, C. Belta, Temporal Logic Motion Planning and Control With Probabilistic Satisfaction Guarantees, IEEE Trans. Robotics 28 (2) (2012) 396–409.

[19] M. Y. Vardi, L. J. Stockmeyer, Improved upper and lower bounds for modal logics of programs, in: Proceedings of the 17th annual ACM symposium on Theory of computing (STOC), ACM, New York, NY, USA, ISBN 0-89791-151-2, 240–251, 1985.

[20] E. A. Emerson, J. Y. Halpern, Decision Procedures and Expressiveness in the Temporal Logic of Branching Time, in: STOC, ACM, 169–180, 1982.

[21] P. Schnoebelen, The complexity of temporal logic model checking, Proceedings of the 4th Advances in Modal Logic 4 (2002) 437–459.

[22] O. Lichtenstein, A. Pnueli, Checking that finite state concurrent programs satisfy their linear specification, in: Proceedings of the 12th ACM SIGACT-SIGPLAN symposium on Principles of programming languages, ACM, 97–107, 1985.

[23] M. Y. Vardi, P. Wolper, An automata-theoretic approach to automatic program verification, in: Proceedings of the First Symposium on Logic in Computer Science, Cambridge, UK, 322–331, 1986.

[24] E. M. Clarke, E. A. Emerson, A. P. Sistla, Automatic verification of finite-state concurrent systems using temporal logic specifications, ACM Transactions on Programming Languages and Systems 8 (2) (1986) 244–263, ISSN 0164-0925.

[25] L. Zhang, U. Hustadt, C. Dixon, A resolution calculus for the branching-time temporal logic CTL, ACM Transactions on Computational Logic (TOCL) 15 (1) (2014) 10.

[26] P. Abate, R. Goré, The Tableau Workbench, Electronic Notes in Theoretical Computer Science 231 (2009) 55–67, ISSN 1571-0661.

[27] R. Goré, J. Thomson, F. Widmann, An Experimental Comparison of Theorem Provers for CTL, in: C. Combi, M. Leucker, F. Wolter (Eds.), Proceedings of the 18th International Symposium on Temporal Representation and Reasoning, IEEE, ISBN 978-1-4577-1242-5, 49–56, 2011.

[28] J. C. M^{c}Cabe-Dansted, A Tableau for RoBCTL*, in: S. Hölldobler, C. Lutz, H. Wansing (Eds.), Proceedings of the 11th European Conference on Logics in Artificial Intelligence (JELIA), vol. 5293, Springer, ISBN 978-3-540-87802-5, 298–310, 2008.

[29] J. C. M^{c}Cabe-Dansted, C. Dixon, CTL-Like Fragments of a Temporal Logic of Robustness, in: N. Markey, J. Wijsen (Eds.), Proceedings of the International Symposium on Temporal Representation and Reasoning, IEEE, ISSN 1530-1311, 11–18, 2010.

[30] G. H. V. Wright, Deontic logic, Mind (1951) 1–15.

[31] D. Nute (Ed.), Defeasible deontic logic, vol. 263, Springer Science & Business Media, 2012.

[32] T. French, J. C. M^{c}Cabe-Dansted, M. Reynolds, A Temporal Logic of Robustness, in: B. Konev, F. Wolter (Eds.), FroCoS, vol. 4720, Springer, ISBN 978-3-540-74620-1, 193–205, 2007.

[33] M. Reynolds, An Axiomatization of Full Computation Tree Logic, The Journal of Symbolic Logic 66 (3) (2001) 1011–1057.

[34] L. Gasparini, T. J. Norman, M. J. Kollingbaum, L. Chen, Severity-Sensitive Robustness Analysis in Normative Systems, in: Revised Selected Papers of the International Workshops on Coordination, Organizations, Institutions, and Norms in Agent Systems X - Volume 9372, Springer, New York, NY, USA, ISBN 978-3-319-25419-7, 72–88, 2015.

[35] McCabe-Dansted, M. Reynolds, To be Fair, Use Bundles, Annals of Mathematics and Artificial Intelligence 80 (3-4) (2017) 72–88.

[36] E. A. Emerson, A. P. Sistla, Deciding Full Branching Time Logic, Information and Control 61 (1984) 175–201.

[37] E. A. Emerson, C. S. Jutla, The Complexity of Tree Automata and Logics of Programs, SIAM Journal on Computing 29 (1) (2000) 132–158, ISSN 0097-5397.

[38] E. A. Emerson, C.-L. Lei, Modalities for model checking (extended abstract): branching time strikes back, in: POPL '85: Proceedings of the 12th ACM SIGACT-SIGPLAN symposium on Principles of programming languages, ACM, New York, NY, USA, ISBN 0-89791-147-4, 84–96, 1985.

[39] A. P. Sistla, E. M. Clarke, The complexity of propositional linear temporal logics, J. ACM 32 (3) (1985) 733–749, ISSN 0004-5411.

[40] H. Hansson, B. Jonsson, A Logic for Reasoning about Time and Reliability, Formal Aspects of Computing 6 (5) (1994) 512–535.

[41] L. D. Alfaro, Temporal logics for the specification of performance and reliability, in: 14th Annual Symposium on Theoretical Aspects of Computer Science (STACS), vol. 1200 of *Lecture Notes in Computer Science*, Springer, 165–176, 1997.

[42] K. D. Heidtmann, Deterministic reliability-modeling of dynamic redundancy, IEEE Transactions on Reliability 41 (3) (1992) 378–385.

[43] M. Bazu, C. Tibeica, L. Galateanu, V. E. Ilian, Fuzzy-Logic Reliability Predictions in Microtechnologies, in: Computational Intelligence for Modelling, Control and Automation, 2005 and International Conference on Intelligent Agents, Web Technologies and Internet Commerce, International Conference on, vol. 1, 89–93, 2005.

[44] G. E. Fainekos, G. J. Pappas, Robustness of temporal logic specifications, Lecture Notes in Computer Science 4262 (2006) 178.

[45] G. E. Fainekos, Robustness of Temporal Logic Specifications, Ph.D. thesis, University of Pennsylvania, 2008.

[46] A. Rizk, G. Batt, F. Fages, S. Soliman, A general computational method for robustness analysis with applications to synthetic gene networks, Bioinformatics 25 (12) (2009) 169–178.

[47] T. Ågotnes, W. van der Hoek, M. Wooldridge, Robust normative systems, in: Proceedings of the 7th international joint conference on Autonomous agents and multiagent systems-Volume 2, International Foundation for Autonomous Agents and Multiagent Systems, 747–754, 2008.

[48] T. Ågotnes, W. V. der Hoek, M. Wooldridge, Robust normative systems and a logic of norm compliance, Logic Journal of IGPL 18 (1) (2010) 4–30.

[49] K. Menger, A logic of the doubtful: On optative and imperative logic, in Reports of a Mathematical Colloquium 2nd series, 2nd issue pages 53–64, 1939.

[50] G.-J. Lokhorst, Mally's Deontic Logic, in: E. N. Zalta (Ed.), The Stanford Encyclopedia of Philosophy, Stanford, winter 2008 edn., 2008.

[51] G.-J. Lokhorst, Ernst Mally's Deontik (1926), Notre Dame Journal of Formal Logic 40 (2) (1999) 273–282.

[52] T. Ågotnes, W. van der Hoek, M. Wooldridge, Logics for Qualitative Coalitional Games, Logic Journal of IGPL 17 (3) (2009) 299.

[53] W. van der Hoek, M. Roberts, M. Wooldridge, Social Laws and Anti-Social Behaviour, in: G. Bonanno, W. van der Hoek, M. Wooldridge (Eds.), Logic and the Foundations of Game and Decision Theory (LOFT 7), vol. 3 of *Texts in Logic and Games*, Amsterdam University Press, 119–152, 2008.

[54] R. Alur, T. A. Henzinger, O. Kupferman, Alternating-time temporal logic, Journal of the ACM 49 (5) (2002) 672–713.

[55] D. Kozen, Results on the Propositional mu-Calculus, Theoretical Computer Science 27 (1983) 333–354.

[56] O. Kupferman, U. Sattler, M. Y. Vardi, The Complexity of the Graded μ-Calculus, in: Automated Deduction - CADE-18, 18th International Conference on Automated Deduction, Copenhagen, Denmark, July 27-30, 2002, Proceedings, 423–437, 2002.

[57] A. Ferrante, M. Napoli, M. Parente, Graded-CTL: Satisfiability and Symbolic Model Checking, in: Formal Methods and Software Engineering, 11th International Conference on Formal Engineering Methods, ICFEM 2009, Rio de Janeiro, Brazil, December 9-12, 2009. Proceedings, 306–325, 2009.

[58] A. Bianco, F. Mogavero, A. Murano, Graded computation tree logic, ACM Transaction Computational Logic 13 (3) (2012) 25:1–25:53.

[59] B. Aminof, A. Murano, S. Rubin, CTL* with graded path modalities, Information and Computation 262 (Part 1) (2018) 1–21.

[60] M. Faella, M. Napoli, M. Parente, Graded alternating-time temporal logic, Fundamenta Informaticae 105 (1-2) (2010) 189–210.

[61] V. Malvone, F. Mogavero, A. Murano, L. Sorrentino, Reasoning about graded strategy quantifiers, Information and Computation 259 (3) (2018) 390–411.

[62] B. Aminof, V. Malvone, A. Murano, S. Rubin, Graded modalities in Strategy Logic, Information and Computation 261 (Part 4) (2018) 634–649.