A Temporal Logic of Robustness

Tim French, John C. M^{c}Cabe-Dansted, and Mark Reynolds

University of Western Australia, Department of Computer Science and Software Engineering
{tim,john,mark}@csse.uwa.edu.au

Abstract. It can be desirable to specify polices that require a system to achieve some outcome even if a certain number of failures occur. This paper proposes a logic, RoCTL*, which extends CTL* with operators from Deontic logic, and a novel operator referred to as "Robustly". This novel operator acts as variety of path quantifier allowing us to consider paths which deviate from the desired behaviour of the system. Unlike most path quantifiers, the Robustly operator must be evaluated over a path rather than just a state; the Robustly operator quantifies over paths produced from the current path by altering a single step. The Robustly operator roughly represents the phrase "even if an additional failure occurs now or in the future". This paper examines the expressivity of this new logic, motivates its use and shows that it is decidable.

Keywords: RoCTL*, Decidability, Modal Logic, Robustness, Branching Time Logic, QCTL*.

1 Introduction

Temporal logic has been particularly useful in reasoning about properties of systems. In particular, the branching temporal logics CTL* [1] and CTL [2] have been used to verify the properties of non-deterministic and concurrent programs.

The logic RoCTL* divides the set of all futures/histories of the CTL* model into successful paths and faulty paths. There is a temporal aspect to faults, so that after the final fault, the path is successful. That is, a path with a finite number of failures has a successful suffix. We augment the operators of CTL* with a Deontic operator, which quantifies over all successful paths, and a novel operator "Robustly" which quantifies over paths which deviate from the current path. Thus there are three path quantifiers in RoCTL*. These quantifiers will be discussed below.

The CTL* "All paths" operator describes hard constraints on the behavior of the system, statements which must be true regardless of how many failures occur. A hard constraint may result from some law of physics, or it may represent something that the system can always be expected to achieve. For example, a real time system may be known to miss some deadlines, but never return an incorrect result. RoCTL* is a conservative extension of LTL and CTL* [3].

To allow us to reason about the consequences of failures, we add an operator "Robustly" ($\blacktriangle$) that allows us to quantify over "deviations" from the current path. For a statement to be robustly true, it must be true on the current path and any path produced by altering a single step. We can represent the statement "if up to n additional failures occur" by chaining n instances of the Robustly operator. To strengthen the meaning of the Robustly operator, we allow the deviating event to be a success as well as a failure. The future after the deviating event may bear no resemblance to the current path. However, to preserve the intuition of the Robustly operator as introducing no more than a single additional failure, no failures occur after the deviating event.

The "Obligatory" operator from Standard Deontic Logic (SDL) is embedded in RoCTL*. This operator is used to describe what the future must be like if no further failures occur. All of the validities of O from SDL hold in RoCTL*. Additionally, as O is used as a path quantifier, all true state formulæ are obligatory in RoCTL*.

The addition of the Robustly operator and temporal operators to Deontic logic allows RoCTL* to deal with Contrary-to-Duty obligations. SDL is able to distinguish what ought to be true from what is true, but is unable to specify obligations that come into force only when we behave incorrectly. For example, SDL is inadequate to represent the obligation "if you murder, you must murder gently" [4]. Addition of temporal operators to Deontic logic allows us to specify correct responses to failures that have occurred in the past [5]. However, this approach alone is not sufficient [5] to represent obligations such as "You must assist your neighbour, and you must warn them iff you will not assist them". In RoCTL* these obligations can be represented if the obligation to warn your neighbour is robust but the obligation to assist them is not.

Other approaches to dealing with Contrary-to-Duty obligations exist. Defeasible logic is often used [6], and logics of agency, such as STIT [7], can be useful as they can allow obligations to be conditional on the agent's ability to carry out the obligation.

This paper provides some examples of robust systems that can be effectively represented in RoCTL*. It is easy to solve the coordinated attack problem if our protocol is allowed to assume that only n messages will be lost. The logic may also be useful to represent the resilience of some economy to temporary failures to acquire or send some resource. For example, a remote mining colony may have interacting requirements for communications, food, electricity and fuel. RoCTL* may be more suitable than Resource Logics (see e.g. [8]) for representing systems where a failure may cause a resource to become temporarily unavailable. This paper presents a simple example where the only requirement is to provide a cat with food when it is hungry.

A number of other extensions of temporal logics have been proposed to deal with Deontic or Robustness issues [9,10,11,12,13]. Each of these logics are substantially different from RoCTL*. Some of these logics are designed specifically to deal with deadlines [9,11]. An Agent Communication Language was formed by adding Deontic and other modal operators to CTL [13]; this language does not explicitly deal with robustness or failures. Hansson and Johnsson [11] proposed an extension of CTL to deal with reliability. However their logic reasons about reliability using probabilities rather than numbers of failures, and their paper does not contain any discussion of the relationship of their logic to Deontic logics. Like our embedding into QCTL*, Aldewereld et al. [12] uses a Viol atom to represent failure. However, their logic also uses probability instead of failure counts and is thus suited to a different class of problems than RoCTL*. Additionally, adding the Viol atom has different expressivity properties to the Robustly operator. CTL* with a special "Viol" atom can express statements such as "If at least one failure occurs" which cannot be expressed in RoCTL*, and it is not known whether all statements that can be expressed in RoCTL* can be trivially translated into CTL*. In particular, it is not known how to translate the phrase "even if a deviation from the current path occurs" into CTL*. None of these logics appear to have an operator that is substantially similar to the Robustly operator of RoCTL*.

Diagnosis problems in control theory [14,15] also deals with failures of systems. Diagnosis is in some sense the dual of the purpose of the RoCTL* logic, as diagnosis requires that failure cause something (detection of the failure) whereas robustness involves showing that failure will *not* cause something.

This paper shows that all RoCTL* statements can be expressed in QCTL*. Furthermore, it is easy to represent statements like "even if n failures occur" in CTL*. However, this paper will show how the RoCTL* logic can represent and make explicit different interactions between the time that failures occur and the time or duration of the effect. There is no known trivial embedding into CTL* that preserves these properties.

2 RoCTL* Logic

2.1 RoCTL* Syntax

RoCTL* extends CTL*, which uses the path operators from LTL:

Next $N\phi$ indicates that ϕ is true at the next step.
Globally $G\phi$ indicates that ϕ is true and will always be true.
Finally $F\phi$ indicates that ϕ will be true at some point in the future.
Until $\phi U\psi$ indicates that ϕ will be true until ψ is true
Weak until $\phi W\psi$ indicates that either $\phi U\psi$ or $G\phi$ is true.

CTL* includes two path-quantifiers:

Always $A\phi$ indicates that ϕ is true in all possible futures.
Exists $E\phi$ indicates that there is a future in which ϕ is true.

RoCTL* Includes the Deontic operators O and P as path-quantifiers.

Obligatory $O\phi$ indicates that in every failure-free future ϕ holds
Permissible $P\phi$ indicates that there is a failure-free future where ϕ holds

RoCTL* has a new pair of path-quantifiers to deal with failures. Unlike A and E which are S5 operators, $\blacktriangle$ is a T operator.

Robustly $\blacktriangle\phi$ indicates that ϕ is true on this path and any path that differs from this path by a single deviating event.

Prone to $\Delta\phi$ indicates that ϕ is true, either on this path or a path differing by a single deviating event, and is the dual of $\blacktriangle$.

The RoCTL* Logic has a set $\mathcal{V}$ of atomic propositions that we call variables. Where p varies over $\mathcal{V}$, the formulæ of RoCTL* are defined by the abstract syntax $\phi := \top \,|\, p \,|\, \neg\phi \,|\, (\phi \wedge \phi) \,|\, (\phi U \phi) \,|\, N\phi \,|\, A\phi \,|\, O\phi \,|\, \blacktriangle\phi$.

The $\top$, $\neg$, $\wedge$, N, U and A are the familiar "true", "not", "and", "next", "until" and "all paths" operators from CTL. The abbreviations $\perp$, $\vee$, F, G, W, E $\rightarrow$ and $\leftrightarrow$ are defined as in CTL* logic. As with SDL logic, we define $P \equiv \neg O \neg$. Finally, we define the abbreviation $\Delta \equiv \neg\blacktriangle\neg$. We say that ϕ is a state formula iff ϕ is equivalent to $A\phi$.

2.2 RoCTL-Structures

Definition 1. *A valuation α is a map from a set of states A to the power set of the variables; we represent the statement "the variable p is true at state w" with $p \in \alpha(w)$.*

Definition 2. *A CTL-structure $M^* = (A^*, \rightarrow^*, \alpha^*)$ is a 3-tuple containing a set of states A^*, a serial (total) binary relation $\rightarrow^*$ and a valuation α^* on the set of states A^*. We define $\mathbb{C}$ as the class of such structures and $\mathbb{C}_t$ as the class of such structures where $\rightarrow^*$ forms a tree.*

Definition 3. *A RoCTL-structure M is a 4-tuple $(A, \xrightarrow{s}, \xrightarrow{f}, \alpha)$, consisting of a set of states A, a serial (total) binary "success" relation $\xrightarrow{s}$, a binary "failure" relation $\xrightarrow{f}$ and a valuation α on the set of states A. We define $\mathbb{M}$ as the class of such RoCTL-structures.*

Definition 4. *We use $\xrightarrow{sf}$ as an abbreviation for $\left(\xrightarrow{s} \cup \xrightarrow{f}\right)$. For all $n \in \mathbb{N}$ we call an ω-sequence $\sigma = \langle w_0, w_1, \ldots\rangle$ of states a fullpath iff for all non-negative integers i we have $w_i \xrightarrow{sf} w_{i+1}$. 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 $\langle w_0, w_1, \ldots, w_i\rangle$.*

Definition 5. *We say that a fullpath σ is failure-free iff for all $i \in \mathbb{N}$ we have $\sigma_i \xrightarrow{s} \sigma_{i+1}$. We define $S\!F(w)$ to be the set of all fullpaths in M starting with w and $S(w)$ to be the set of all* failure-free *fullpaths in M starting with w.*

Definition 6. *For two fullpaths σ and π we say that π is an i-deviation from σ iff $\sigma_{\leq i} = \pi_{\leq i}$ and $\pi_{\geq i+1} \in S(\pi_{i+1})$. 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 a fullpath to a set of fullpaths such that where σ and π are fullpaths, π is a member of $\delta(\sigma)$ iff π is a deviation from σ.*

Below is an example of an i-deviation π from a fullpath σ. The arrows not labeled with s can be either $\xrightarrow{s}$ or $\xrightarrow{f}$. After π diverges from σ, it avoids any failures that may have been on $\sigma_{>i}$. We require that a deviation not introduce any failures except for the deviating event itself, hence $\pi_{\geq i+1}$ is failure-free.

$$\begin{tikzcd}
& & & \pi_{i+1} \arrow[r, "s"] & \pi_{i+2} \arrow[r, "s"] & \pi_{i+3} \arrow[r, dotted, "s\cdots"] & {} \\
\sigma_0 \arrow[r] & \sigma_1 \arrow[r, dotted] & \sigma_i \arrow[r] \arrow[rru] & \sigma_{i+1} \arrow[r] & \sigma_{i+2} \arrow[r] & \sigma_{i+3} \arrow[r, dotted] & {}
\end{tikzcd}$$

Note that after the deviation, all transitions must be success transitions while steps before the deviation may be either success or failure transitions. This follows from the intuition of Robustly representing "even if an additional failure occurs", as the failures that are prior to the deviation are already on the existing path. Additionally, allowing failures to occur before the deviation also allows n Robustly operators to be nested to represent the statement "even if n additional failures occur".

2.3 RoCTL* Semantics

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

$$M, \sigma \vDash N\phi \text{ iff } M, \sigma_{\geq 1} \vDash \phi$$
$$M, \sigma \vDash \phi U \psi \text{ iff } \exists_{i \in \mathbb{N}} \text{ s.t. } M, \sigma_{\geq i} \vDash \psi \text{ and } \forall_{j \in \mathbb{N}} j < i \implies M, \sigma_{\geq j} \vDash \psi$$
$$M, \sigma \vDash A\phi \text{ iff } \forall_{\pi \in \mathcal{SF}(\sigma_0)} M, \pi \vDash \phi$$
$$M, \sigma \vDash O\phi \text{ iff } \forall_{\pi \in S(\sigma_0)} M, \pi \vDash \phi$$
$$M, \sigma \vDash \blacktriangle\phi \text{ iff } \forall_{\pi \in \delta(\sigma)} M, \pi \vDash \phi \text{ and } M, \sigma \vDash \phi\,.$$

The definitions for $\top$, p, $\neg$ and $\wedge$ are as we would expect from classical logic. We say that a formula ϕ is valid in RoCTL* iff for all structures M in $\mathbb{M}$, for all fullpaths σ in M we have $M, \sigma \vDash \phi$.

The operator O is similar to A. If ψ is a CTL* formula and ψ' is ψ with all instances of A replaced with O, then $O\psi'$ is a validity of RoCTL* iff ψ is a validity [3]. It is also easy to show that if $\xrightarrow{f}$ is empty, the O and $\blacktriangle$ operators are equivalent to the A operator. Restricting the class of structures such that $\xrightarrow{s} \cap \xrightarrow{f} = \emptyset$ and/or such that $\xrightarrow{f}$ is serial does not affect the validities of the logic.

3 Properties of RoCTL*

The behaviour of the A operator is the same as in CTL*, and the behaviour of the O operator is similar to that of the O operator in SDL. It is easy to show [3] that the axiom class $O\phi \rightarrow P\phi$ is valid in RoCTL* (and SDL); neither the axiom class $\phi \rightarrow P\phi$, nor the axiom class $O\phi \rightarrow \phi$, is valid in RoCTL* (or SDL). However, unlike SDL, the axiom class $p \rightarrow Op$ is valid in RoCTL* [3]. This is due to O being a path quantifier in RoCTL*. In this section, combinations of operators and differences between the operators will be discussed.

3.1 Interpretations of Combinations of Operators

As both A and O are traditional path-quantifiers, AO and OA are of little use and reduce to O and A respectively. Similarly, $\blacktriangle O$ reduces to O. However, $O\blacktriangle$ does not reduce to $\blacktriangle$. Indeed, $\blacktriangle\phi$ is not a state formula, while $O\blacktriangle\phi$ is. The pair $O\blacktriangle$ quantifies over any path starting at the present state that has at most one failure.

The order of $\blacktriangle$ and N affects the meaning of the combination, $\blacktriangle N\phi$ represents the statement "Even if one or fewer deviations occur now or in the future, ϕ will be true at the next step". The formula $N\blacktriangle\phi$ is similar, but only requires ϕ to be true if the deviation occurs after the next step. As with the A path-quantifier from CTL*, $\blacktriangle N\phi \rightarrow N\blacktriangle\phi$ and $\blacktriangle G\phi \rightarrow G\blacktriangle\phi$ are valid in RoCTL* but $\blacktriangle N\phi \leftarrow N\blacktriangle\phi$ and $G\blacktriangle\phi \rightarrow \blacktriangle G\phi$ are not [3]. The statement $G\blacktriangle\phi$ indicates that for every state along the present path, ϕ holds at the beginning of all deviations; the statement $\blacktriangle G\phi$ indicates that ϕ will be true not only at the beginning of the deviation, but along the deviation as well.

The order of the N and O operators is important, even more so than of N and A. As with $NA\phi \rightarrow AN\phi$, the formula form $NO\phi \rightarrow ON\phi$ is not valid. However, unlike $AN\phi \rightarrow NA\phi$, the form $ON\phi \rightarrow NO\phi$ is not valid. The combination NO represents what will obligatory after the next transition, which might be a failure transition. The ON combination instead discusses what should be true at the next step. For further discussion of this combination, see Example 1.

It is easy to show that the following formulæ forms are valid in RoCTL*: $A\phi \rightarrow O\phi$, $AO\phi \leftrightarrow O\phi$, $OA\phi \leftrightarrow A\phi$, $A\phi \rightarrow \blacktriangle\phi$, $\blacktriangle\phi \rightarrow O\phi$, $A\blacktriangle\phi \leftrightarrow A\phi$, $\blacktriangle A\phi \leftrightarrow A\phi$ and $\blacktriangle O\phi \leftrightarrow O\phi$.

3.2 Differences Between A, $\blacktriangle$ and O

The A, $\blacktriangle$ and O operators have similar properties since they quantify over paths. The $\blacktriangle$ operator is an unusual path quantifier as $\blacktriangle\phi$ is not a state formula. This paper will not present a full axiomatization of RoCTL*. However, it will examine which axioms of A are also valid for $\blacktriangle$ and O . The axioms [16] that reference the A operator are:

C9	$A(\phi \rightarrow \psi) \rightarrow (A\phi \rightarrow A\psi)$	C13	$A\neg\phi \leftrightarrow \neg E\phi$
C10	$A\phi \rightarrow AA\phi$	C14	$p \rightarrow Ap$
C11	$A\phi \rightarrow \phi$	C15	$AN\phi \rightarrow NA\phi$
C12	$\phi \rightarrow AE\phi$		

LC $\quad AG(A\phi \rightarrow EN(A\psi U A\phi)) \rightarrow (A\phi \rightarrow EG(A\psi U A\phi))$.

As the real world may not be in a desirable state, neither $O\phi \rightarrow \phi$ nor $\phi \rightarrow OP\phi$ are valid. As mentioned previously $ON\phi \rightarrow NO\phi$ is not valid in RoCTL*. It is easy to show that the C9, C10, C13, C14 and LC axioms are still valid if A is replaced with O.

The formula $\phi \rightarrow \blacktriangle\Delta\phi$ is not valid in RoCTL*. The reason for this is that a deviation can prevent any number of failures occurring in the future, but the second deviation can cause only a single new failure. The $\blacktriangle$ operator is not transitive, so $\blacktriangle\phi \rightarrow \blacktriangle\blacktriangle\phi$ is not valid in RoCTL*. The axiom forms C9, C13, C14, C15 are still valid in RoCTL* if A is replaced with $\blacktriangle$. The formula $\blacktriangle G\left(A\phi \rightarrow PN\left(A\psi UA\phi\right)\right) \rightarrow \left(A\phi \rightarrow \Delta G\left(A\psi UA\phi\right)\right)$ is also valid in RoCTL*.

4 Examples

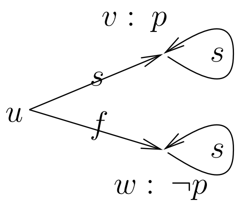
In this section a number of examples are presented. The first example examines the difference between the formula $NO\phi$ and the formula $ON\phi$. The second example shows how RoCTL* may be used to specify a robust network protocol. Then an example of feeding a cat will be introduced to explain how we may reason about consequences of polices. These examples will frequently use the $\blacktriangle/\Delta$ operator to form the pair $O\blacktriangle$. In the final example we use the simple formula $O(\Delta Fe \rightarrow Fw)$ which nests $\blacktriangle/\Delta$ in a less trivial way.

Example 1. Here is an example of a simple Contrary-to-Duty obligation. This provides a counter example to both $ON\phi \rightarrow NO\phi$ and $NO\phi \rightarrow ON\phi$.

$ON(Gp)$**:** You should commit to the proper decision.
$NO\left(G\neg p \vee Gp\right)$**:** Once you have made your decision, you must stick with it.

It is consistent with the above that we do not make the proper decision ($N\neg p$). Once we have made the wrong decision we cannot satisfy Gp, so we must stick with the wrong decision $G\neg p$. Hence, in this case, both $ON(Gp)$ and $NO(G\neg p)$ are true. Likewise $ON(G\neg p)$ and $NO(Gp)$ are false. This demonstrates how obligations can change with time in RoCTL*. We will now give an example of a structure $M = (A, \xrightarrow{s}, \xrightarrow{f}, \alpha)$ that satisfies these formulae:

$$\begin{aligned} A &= \{u, v, w\}, \\ \xrightarrow{s} &= \left\{(u,v), (v,v), (w,w)\right\}, \\ \xrightarrow{f} &= \left\{(u,w)\right\}, \\ \alpha(v) = \{p\}, &\quad \alpha(w) = \emptyset\,. \end{aligned}$$



$$\begin{tikzcd} & v : p \arrow[loop right, "s"] \\ u \arrow[ur, "s"] \arrow[dr, "f"] & \\ & w : \neg p \arrow[loop right, "s"] \end{tikzcd}$$

Let σ be the fullpath $\langle u, w, w, \ldots\rangle$ corresponding to making the wrong decision. We see that $M, \sigma_{\geq 1} \vDash \neg p$, so $M, \sigma_{\geq 1} \vDash O\neg p$ and $M, \sigma_{\geq 1} \vDash \neg Op$. Thus $M, \sigma \vDash NO\neg p$ and $M, \sigma \vDash N\neg Op$. It follows that $M, \sigma \vDash \neg NOp$.

Let $\pi = \langle v, v, \ldots\rangle$. We see that $M, \pi \vDash p$. We see that $S(u) = \{\langle u, v, v, \ldots\rangle\}$. Hence $M, \sigma \vDash ONp$ and it follows that $M, \sigma \vDash \neg O\neg Np$ and so $M, \sigma \vDash \neg ON\neg p$.

Hence $M, \sigma \vDash (ONp \wedge \neg NOp)$ and so $M, \sigma \nvDash (ON\phi \rightarrow NO\phi)$ where $\phi = p$. Likewise $M, \sigma \vDash (NO\neg p \wedge \neg ON\neg p)$, so $M, \sigma \nvDash (NO\phi \rightarrow ON\phi)$ where $\phi = \neg p$.

Example 2. In the coordinated attack problem we have two generals A and B. General A wants to organise an attack with B. A communication protocol will be presented such that a coordinated attack will occur if no more than one message is lost.

$AG\left(s_A \rightarrow ONr_B\right)$**:** If A sends a message, B should receive it at the next step.
$AG\left(\neg s_A \rightarrow \neg Nr_B\right)$**:** If A does not send a message now, B will not receive a message at the next step.

$AG(f_A \rightarrow AGf_A)$**:** If A commits to an attack, A cannot withdraw.
$AG(f_A \rightarrow \neg s_A)$**:** If A has committed to an attack, it is too late to send messages.
$A\,(\neg f_A W r_A)$**:** A cannot commit to an attack until A has received plans from B

Similar constraints to the above also apply to B. Below we add a constraints requiring A to be the general planning the attack

$A\,(\neg s_B W r_B)$**:** General B will not send a message until B has received a message.

No protocol exists to satisfy the original coordination problem, since an unbounded number of messages can be lost. Here we only attempt to ensure correct behaviour if one or fewer messages are lost.

$A\,(s_A U r_A)$**:** General A will send plans until a response is received.
$AG\,(r_A \rightarrow f_A)$**:** Once general A receives a response, A will commit to an attack.
$A\,(\neg r_B W\,(r_B \wedge (s_B \wedge N s_B \wedge NN f_B)))$**:** Once general B receives plans, B will send two messages to A and then commit to an attack.

Having the formal statement of the policy above and the semantics of RoCTL* we may prove that the policy $\hat{\phi}$ is consistent and that it implies correct behaviour even if a single failure occurs:

$$\hat{\phi} \rightarrow O\blacktriangle F(f_A \wedge f_B)\,.$$

Indeed, we will show in Section 5 that such issues can be decided in finite time.

Example 3. 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. We have the following variables:

b "The cat bowl is full at the beginning of this hour"
d "This hour is feeding time"

We can translate the statements above into RoCTL* statements:

1. $AG(d \rightarrow \neg Nd)$: If this hour is feeding time, the next is not.
2. $AG((d \vee \neg b) \rightarrow \Delta N \neg b)$: If it is feeding time or the cat bowl was empty, a single failure may result in an empty bowl at the next step
3. $AG((\neg d \wedge b) \rightarrow Nb)$: If the bowl is full and it is not feeding time, the bowl will be full at the beginning of the next hour.
4. $O\blacktriangle G\,(d \rightarrow b)$: It is obligatory that, even if a single failure occurs, it is always the case that the bowl must be full at feeding time.
5. b: The cat bowl starts full.

Having the formalised the policy it can be proven that the policy is consistent and that the policy implies $O\blacktriangle GONb$, indicating that the bowl must be filled at every step (in case we forget at the next step), unless we have already failed twice. The formula $AGONb \rightarrow O\blacktriangle G\,(d \rightarrow b)$ can also be derived, indicating that following a policy requiring us to always attempt to fill the cat bowl ensures that we will not starve the cat even if we make a single mistake. Thus following this simpler policy is sufficient to discharge our original obligation.

Example 4. We define a system that will warn the user if the system enters an unsafe state:

1. $AGONs$: The system should always ensure that the system reaches a safe state by the next step.
2. $AG(s \rightarrow N\neg e)$: 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. $AG(\neg s \rightarrow Nw)$: If the system is in an unsafe state, the system will warn the user at the next step.

We may prove that if an error e almost occurs, the system will finally warn the user, i.e. $O(\Delta Fe \rightarrow Fw)$.

See [3] for more examples, such as nesting Δ within $\blacktriangle$ to discuss liveness of failure detection.

5 Embeddings

In this section it will be shown that RoCTL* can be embedded in QCTL*, and hence is decidable.

5.1 Converting a RoCTL-Structure M to a CTL-Structure M^*

For a RoCTL-structure M we construct a CTL tree structure M^* as follows. We define an interim CTL-structure M^{sf}. The atoms of M^{sf} are the atoms of M plus an additional atom "Viol" representing the statement "The last transition was a failing (f) transition" and an additional set Y of anonymous atoms. For each state w of M we have two states w^s and w^f in M^{sf}, with Viol being true at w^f but not w^s. For each pair of states w_i and w_j in M, we have

$$\begin{array}{ll} w_i^s \rightarrow w_j^s \iff w_i \xrightarrow{s} w_j & \quad w_i^s \rightarrow w_j^f \iff w_i \xrightarrow{f} w_j \\ w_i^f \rightarrow w_j^s \iff w_i \xrightarrow{s} w_j & \quad w_i^f \rightarrow w_j^f \iff w_i \xrightarrow{f} w_j \,. \end{array}$$

It is easy to rewrite a RoCTL* formula to use the Δ operator instead of $\blacktriangle$. To ease the embedding of the Δ operator we will add a countable set Y of atoms to M^{sf} and unwind it into a tree to form the tree structure M^*.

Translating a CTL tree structure $M^* = (A^*, \rightarrow^*, \alpha^*)$ with a Viol atom into a RoCTL-structure $M = (A, \xrightarrow{s}, \xrightarrow{f}, \alpha)$ is trivial. We set $A = A^*$ and let $\alpha^* = \alpha$. We set $\xrightarrow{s}$ and $\xrightarrow{f}$ such that for each pair of states w_i and w_j we have:

$$\begin{aligned} w_i \xrightarrow{s} w_j &\iff (w_i \rightarrow^* w_j) \wedge (\text{Viol} \notin \alpha(w_j)) \\ w_i \xrightarrow{f} w_j &\iff (w_i \rightarrow^* w_j) \wedge (\text{Viol} \in \alpha(w_j)) \,. \end{aligned}$$

5.2 Translating a RoCTL* Formula into a QCTL* Formula

Definition 7. *Given some CTL structure $M = (A, \rightarrow, \alpha)$ and some $x \in A$, an* *x-variant* *of M is some structure $M = (A, \rightarrow, \alpha')$ where $\alpha'(w) \backslash \{x\} = \alpha(w) \backslash \{x\}$ for all $w \in A$.*

QCTL* has the syntax $\phi := \top \,|\, p \,|\, \neg\phi \,|\, (\phi \wedge \phi) \,|\, (\phi U \phi) \,|\, N\phi \,|\, A\phi \,|\, \exists_p \phi$. The semantics of $\top$, p, $\neg$, $\wedge$, U, N, and A are the same as in CTL* and RoCTL*. Under the Kripke semantics for QCTL*, $\exists_p \phi$ is defined as

$$M, b \models \exists_p \alpha \iff \text{There is some } p\text{-variant } M' \text{ of } M \text{ such that } M', b \models \alpha \,.$$

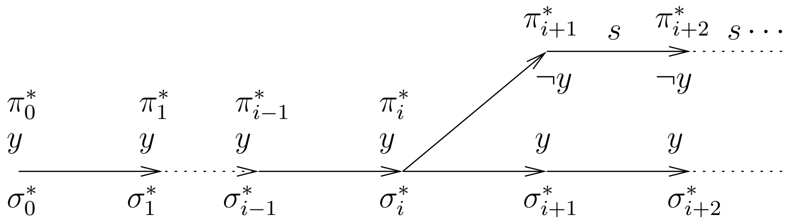
This paper uses the tree semantics for QCTL*. These semantics are the same as the Kripke semantics except that, whereas the Kripke semantics evaluates validity over the class $\mathbb{C}$ of CTL-structures, the tree semantics evaluate validity over the class $\mathbb{C}_t$ of tree CTL-structures. This changes the validities of the logic as, unlike CTL* [17], QCTL* is sensitive to unwinding into a tree structure [18].

We let γ be the (Q)CTL* formula $NNG\neg$Viol. The γ formula is used to represent the requirement that all transitions after a deviation must be successes.

We define a translation function t^{Δ} such that for any formula ϕ^* and for some atom y not in ϕ^*:

$$t^{\Delta}(\phi^*) = \forall_y \left[Gy \rightarrow E\left[(Gy \vee F(y \wedge \gamma)) \wedge \phi^* \right] \right] \,.$$

Note that for $t^{\Delta}(\phi^*)$ to hold, $E\left[(Gy \vee F(y \wedge \gamma)) \wedge \phi^*\right]$ must hold for all possible values of y that satisfy Gy, including the case where y is true only along the current fullpath σ^*. The diagram below shows a fullpath π^* that satisfies $F(y \wedge \gamma)$ for all such y.



Lemma 1. *Say that ϕ is a RoCTL* formula and ϕ^* is a QCTL* formula such that for all M and σ it is the case that $M, \sigma \models \phi$ iff $M^*, \sigma^* \models \phi^*$. Then, for all M and σ it is the case that $M, \sigma \models \Delta\phi$ iff $M^*, \sigma^* \models t^{\Delta}(\phi^*)$.*

Proof. ($\Longrightarrow$) Say that $M, \sigma \models \Delta\phi$. Then $M, \sigma \models \phi$ or there exists a deviation π from σ such that $M, \pi \models \phi$. If $M, \sigma \models \phi$ then $M^*, \sigma^* \models \phi^*$ from which it follows that $M^*, \sigma^* \models \forall_y \left[Gy \rightarrow E\left[Gy \wedge \phi^* \right] \right]$, and so $M^*, \sigma^* \models t^{\Delta}(\phi^*)$.

If $M, \sigma \not\models \phi$ then, for some i, there exists an i-deviation π from σ such that $M, \pi \models \phi$. If Gy holds along σ^* then y holds at $\pi^*_i = \sigma^*_i$. As π is an i-deviation, all transitions following π_{i+1} are success transitions, so $M^*, \pi^*_{\geq i} \models \gamma$ and $M^*, \pi^* \models F(y \wedge \gamma) \wedge \phi^*$ from which it follows that $M^*, \sigma^* \models t^{\Delta}(\phi^*)$.

($\Longleftarrow$) Say that $M^*, \sigma^* \vDash t^{\Delta}(\phi^*)$. Then

$$M^y, \sigma^* \vDash [Gy \rightarrow E[(Gy \vee F(y \wedge \gamma)) \wedge \phi^*]] ,$$

where M^y is any tree structure that is y-bisimilar to M^*. Consider an M^y for which y is true at a state w iff $w \in \sigma^*$. Then $M^y, \sigma^* \vDash E[(Gy \vee F(y \wedge \gamma)) \wedge \phi^*]$. Thus there exists some fullpath σ^y such that $\sigma^y_0 = \sigma^*_0$ and $M^y, \sigma^y \vDash F(y \wedge \gamma) \wedge \phi^*$ or $M^y, \sigma^y \vDash Gy \wedge \phi^*$.

If $M^y, \sigma^y \vDash Gy \wedge \phi^*$ then $\sigma^y = \sigma^*$, so $M^*, \sigma^* \vDash \phi^*$ and $M, \sigma \vDash \phi$. If $M^y, \sigma^y \vDash F(y \wedge \gamma) \wedge \phi^*$ then there exists an integer i such that $M^y, \sigma^y_{\geq i} \vDash y \wedge \gamma$. Let σ and π be translations of σ^* and σ^y respectively into fullpaths through the original structure M. As M^y is a tree structure and y is only true along σ^*, it follows that $\sigma^y_{\leq i} = \sigma^*_{\leq i}$ and $\pi_{\leq i} = \sigma_{\leq i}$. This, together with the fact that $M^y, \sigma^y_{\geq i} \vDash \gamma$, means that π is an i-deviation from σ. As $M^y, \sigma^y \vDash \phi^*$ it follows that $M, \pi \vDash \phi$, and so $M, \sigma \vDash \Delta\phi$.

Theorem 1. *We may express any RoCTL* formula ϕ of length n as a QCTL* formula ϕ^* of length $\mathcal{O}(n)$ that is equivalent to ϕ when ϕ^* is interpreted according to the tree semantics for QCTL*.*

Proof. Using $t^{\Delta}(\phi^*)$ defined above, $t^{O}(\phi^*) \equiv A(NG\neg\text{Viol} \rightarrow \phi^*)$, $t^{\neg}(\phi^*) \equiv \neg\phi^*$, $t^{N}(\phi^*) \equiv N\phi^*$, $t^{A}(\phi^*) \equiv A\phi^*$, $t^{\wedge}(\phi^*_1, \phi^*_2) \equiv \phi^*_1 \wedge \phi^*_2$, $t^{U}(\phi^*_1, \phi^*_2) \equiv \phi^*_1 U \phi^*_2$, $t^{\top} = \top$ and $t^{p}(p) \equiv p$, we may recursively translate any RoCTL* formula ϕ into a QCTL* formula ϕ^* such that $M, \sigma \vDash \phi$ iff $M^*, \sigma^* \vDash \phi^*$ where M^*, σ^* are the transformations of M, σ described above.

Corollary 1. *RoCTL* is decidable.*

Proof. In Section 5.1 we have shown that for every RoCTL-structure M there is a corresponding CTL tree structure M^* and visa versa. As the tree semantics for QCTL* are decidable [19,20], it is obvious from Theorem 1 that RoCTL* is decidable.

6 Conclusion

We have proposed a logic RoCTL* for reasoning about robust systems. This logic introduced a Robustly operator that provides a bridge between what should happen and what actually does. We have given examples of simple robust systems that can be represented in RoCTL*. We have proven that RoCTL* has a linear embedding into QCTL*, and hence is decidable. Never-the-less there is much more to be understood about this logic.

Although we can decide RoCTL* via QCTL*, it is important to find a more efficient decision procedure as QCTL* is not elementary [21,22]. Determining whether a particular model satisfies a RoCTL* policy is also useful. It is easy to show that the model checking problem is decidable via reduction to QCTL* and μ-calculus. However, a more efficient decision procedure is needed.

Restricting the expressivity of RoCTL* may lead to better complexity results. It may be productive to look for a restriction of RoCTL* that more closely resembles CTL or LTL than CTL*. However, as the Robustly operator is a path-quantifier that is not a state formulae, it is quite different from the operators found in CTL and LTL. For this reason, finding a restriction that preserves the usefulness and uniqueness of the Robustly operator while gaining some of the simplicity of CTL or LTL will be non-trivial.

Finding an axiomatization for RoCTL* may be a challenging task. Although this paper has compared the validities of the $\blacktriangle$ and O operators to the axioms for the A operator, we are far from finding a sound axiomatization of RoCTL*, and have not examined how to prove that such an axiomatization is complete.

More work needs to be done in applying RoCTL* to practical problems. This paper has presented some trivial examples where RoCTL* succinctly represents robustness properties of simple systems. To test the expressivity of RoCTL* and find real world applications, much larger and more complex examples need to be formalised and examined. For some uses, RoCTL* may need to be extended. RoCTL* can use the prone operator to discuss whether it is possible for a failure to be detected at a particular step. Diagnosis problems require that failures *will* be detected. For these purposes, an "If *at least* one additional failure occurs" operator and a knowledge operator are desirable. It would be useful to find an extension that satisfies these requirements while preserving decidability.

References

1. Emerson, E.A., Sistla, A.P.: Deciding full branching time logic. Technical report, University of Texas at Austin, Austin, TX, USA (1985)
2. Clarke, E., Emerson, E.: Synthesis of synchronization skeletons for branching time temporal logic. In: Proc. IBM Workshop on Log. of Progr., Yorktown Heights, pp. 52–71. Springer, Heidelberg (1981)
3. French, T., M^{c}Cabe-Dansted, J.C., Reynolds, M.: A temporal logic of robustness, RoCTL*. Technical report, UWA (2007) `http://dansted.org/RoCTL07.pdf`
4. Forrester, J.W.: Gentle murder, or the adverbial samaritan. J. Philos. 81(4), 193–197 (1984)
5. van der Torre, L.W.N., Tan, Y.: The temporal analysis of Chisholm's paradox. In: Senator, T., Buchanan, B. (eds.) Proc. 14th Nation. Conf. on AI and 9th Innov. Applic. of AI Conf., Menlo Park, California, pp. 650–655. AAAI Press, Stanford, California (1998)
6. McCarty, L.T.: Defeasible deontic reasoning. Fundam. Inform. 21(1/2), 125–148 (1994)
7. Belnap, N.: Backwards and forwards in the modal logic of agency. Philos. Phenomen. Res. 51(4), 777–807 (1991)
8. de Weerdt, M., Bos, A., Tonino, H., Witteveen, C.: A resource logic for multi-agent plan merging. Annals of Math. and AI 37(1-2), 93–130 (2003)
9. Broersen, J., Dignum, F., Dignum, V., Meyer, J.J.C.: In: Designing a Deontic Logic of Deadlines. In: Lomuscio, A.R., Nute, D. (eds.) DEON 2004. LNCS (LNAI), vol. 3065, pp. 43–56. Springer, Heidelberg (2004)
10. Long, W., Sato, Y., Horigome, M.: Quantification of sequential failure logic for fault tree analysis. Reliab. Eng. Syst. Safe. 67, 269–274 (2000)

11. Hansson, H., Jonsson, B.: A logic for reasoning about time and reliability. Form. Asp. Comput. 6(5), 512–535 (1994)
12. Aldewereld, H., Grossi, D., Vazquez-Salceda, J., Dignum, F.: Designing normative behaviour by the use of landmarks. In: Agents, Norms and Institutions for Regulated Multiag. Syst., Utrecht, The Netherlands (2005)
13. Rodrigo, A., Eduardo, A.: Normative pragmatics for agent communication languages. In: Akoka, J., Liddle, S.W., Song, I.-Y., Bertolotto, M., Comyn-Wattiau, I., van den Heuvel, W.-J., Kolp, M., Trujillo, J., Kop, C., Mayr, H.C. (eds.) Perspectives in Conceptual Modeling. LNCS, vol. 3770, pp. 172–181. Springer, Heidelberg (2005)
14. Jéron, T., Marchand, H., Pinchinat, S., Cordier, M.O.: Supervision patterns in discrete event systems diagnosis. In: 8th Internat. Workshop on Discrete Event Syst., pp. 262–268 (2006)
15. Arnold, A., Vincent, A., Walukiewicz, I.: Games for synthesis of controllers with partial observation. TCS 303(1), 7–34 (2003)
16. Reynolds, M.: An axiomatization of full computation tree logic. J. Symb. Log. 66(3), 1011–1057 (2001)
17. Emerson, E.A.: Alternative semantics for temporal logics. TCS 26, 121–130 (1983)
18. Kupferman, O.: Augmenting branching temporal logics with existential quantification over atomic propositions. In: Comput. Aid. Verfic., Proc. 7th Int. Conf., Liege, pp. 325–338. Springer, Heidelberg (1995)
19. Emerson, E.A., Sistla, A.P.: Deciding branching time logic. In: STOC '84: Proc. 16^{th} annual ACM sympos. on Theory of computing, New York, NY, USA, pp. 14–24. ACM Press, New York (1984)
20. French, T.: Decidability of quantifed propositional branching time logics. In: AI '01. Proc. 14^{th} Austral. Joint Conf. on AI, London, UK, pp. 165–176. Springer, Heidelberg (2001)
21. Sistla, A.P., Vardi, M.Y., Wolper, P.: The complementation problem for bučhi automata with applications to temporal logic. TCS 49(2-3), 217–237 (1987)
22. French, T.: Bisimulation Quantifiers for Modal Logics. PhD thesis, UWA (2006)