CSSE Technical Report UWA-CSSE-07-003
May 2007
A Temporal Logic of Robustness

John C. M^{c}Cabe-Dansted and Tim French and Mark Reynolds
School of Computer Science & Software Engineering
The University of Western Australia
Crawley WA 6009 Australia
email: {john,tim,mark}@csse.uwa.edu.au
tel: +61 8 6488 2279
fax: +61 8 6488 1089

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 represents the phrase "even if an additional failure occurs now or in the future", and thus the paths quantified over depend upon the failures already on the path. 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* [8] and CTL [5] have been used to verify the properties of non-deterministic and concurrent programs.

We frequently wish to specify that some temporal property will hold even if the system suffers from a certain number of failures. A common example is an error-correcting code which will ensure that a signal is sent correctly provided the number of transmission errors is below some fixed amount. We will define a logic for reasoning about the effect of a limited number of failures, and for specifying how well the system must respond to a certain number of failures.

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 by at most one fault. 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*.

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*.

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 "even if 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.

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. [6]) 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.

Deontic logic has many paradoxes. Some of these, such as the "Gentle Murderer" paradox spring from the inadequacy of Standard Deontic Logic for dealing with obligations caused by acting contrary to duty such as "If you murder, you must murder gently". Contrary-to-duty obligations are important for modeling a robust system, as it is often important to state that the system should achieve some goal and also that if it fails it should in some way recover from the failure. Defeasible logic is often used to deal with contrary-to-duty obligations [17]. Logics of agency, such as STIT [3], can be useful as they can allow obligations to be conditional on the agent's ability to carry out the obligation. Another approach is to add temporal operators to Deontic logic, so that we can deal with futures that exhibit correct responses to failures that have occurred in the past [21]; RoCTL* follows this approach. However, this approach alone is not sufficient [21] 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. Contrary-to-duty obligations have interesting effects on RoCTL* logic. For example, the statement "it is obligatory for ϕ to be true at the next step" neither implies nor is implied by the statement "At the next step it is obligatory for ϕ to be true" (see Example 1).

Diagnosis problems in control theory [14,2] deal with detecting failures in a system. This 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 undesirable.

A number of other extensions of temporal logics have been proposed to deal with Deontic or Robustness issues [4,16,13,1,19]. Each of these logics are substantially different from RoCTL*. Some of these logics are designed specifically

to deal with deadlines [4,13]. An Agent Communication Language was formed by adding Deontic and other modal operators to CTL [19]; this language does not explicitly deal with robustness or failures. Hansson and Johnsson [13] proposed an extension of CTL to deal with reliability. However, as well as being intended to deal with deadlines, 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. [1] 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*.

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

We may represent the statement "Even if n or fewer unexpected events occur" by chaining together n instances of the ▲ operator.

The RoCTL* Logic has a set $\mathcal{V}$ of atomic propositions that we call variables. The formulæ of RoCTL* are defined by the following abstract syntax where p varies over $\mathcal{V}$:

$$\phi := \top | p | \neg\phi | (\phi \wedge \phi) | (\phi U \phi) | N\phi | A\phi | O\phi | \blacktriangle\phi$$

The $\top$, $\neg$, $\wedge$, N, U and A are the familiar "true", "not", "and", "next", "until" and "all paths" operators from CTL. The abbreviations $\bot$, $\vee$, 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 $\vartriangle \equiv \neg\blacktriangle\neg$. 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 worlds A to the power set of the variables; we represent the statement "the variable p is true at world 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 worlds 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 worlds A. We define $\mathbb{M}$ as the class of such RoCTL-structures.*

Note 1. We may assume, without loss of generality, that $\xrightarrow{s} \cap \xrightarrow{f} = \emptyset$.

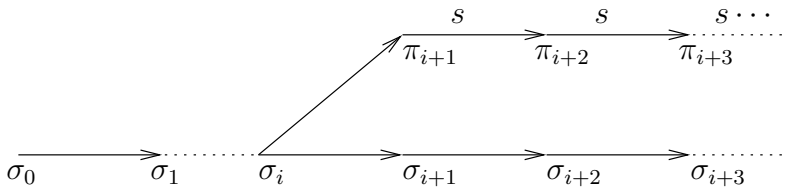
Definition 4. *We use $\xrightarrow{sf}$ as an abbreviation for $\left(\xrightarrow{s} \cup \xrightarrow{f}\right)$.*

Definition 5. *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 6. *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 $SF(w)$ 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 7. *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 $\xrightarrow{f}$. The diagonal arrow represents the unexpected event, which can be a success or failure. 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.



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:

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

The definition for $\top$, p, $\neg$ and $\wedge$ is as we would expect from classic 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 $M, \sigma \vDash \phi$.

Our logic is designed to reason about failures that are in some sense chaotic. The intended behaviour of the system, and the structure M may have some finite number of repeating states. However, the real world can be chaotic and, in effect, has an infinite number of states no two of which are entirely equivalent; then each decision made within what is in essence a tree structure. Each failure is activated by reaching one particular state of this tree. If a deviation from the current path occurs before reaching that state, then that state will never be reached.

3 Properties of RoCTL*

3.1 Interpretations of Combinations of Operators

As both A and O are path-quantifiers, AO and OA are of little use and reduce to O and A respectively.

The order of $\blacktriangle$ and N affects the meaning of the combination.

$\blacktriangle N$ Even if a one or fewer deviations occur now or in the future, it will be true at the next step that ...

$N\blacktriangle$ At the next step, if a one or fewer deviations occur now or in the future, it will be true that ... /If one or fewer deviations occur in the future ....

The pair $\blacktriangle N$ is similar to the pair $N\blacktriangle$, except that $N\blacktriangle$ does not consider paths where the deviation occurs on the first step. As with the A operator in CTL*, $\blacktriangle N\phi \rightarrow N\blacktriangle\phi$ is valid in RoCTL* but $\blacktriangle N\phi \leftarrow N\blacktriangle\phi$ is not [12].

$\blacktriangle G$ Even if one or fewer additional failures occur, it will always be true that ...
$G\blacktriangle$ In this future, it will always be the case that ... even if one or fewer additional failures occurs.

The difference between $G\blacktriangle$ and $\blacktriangle G$ is examined further in Example 6. As with the A operator in CTL*, $\blacktriangle G\phi \rightarrow G\blacktriangle\phi$ is valid in RoCTL* but $G\blacktriangle\phi \rightarrow \blacktriangle G\phi$ is not. The statement $G\blacktriangle\phi$ indicates that for every point along the present fullpath, 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 A, the formula form $NO\phi \rightarrow ON\phi$ is not valid. However, unlike A the form $ON\phi \rightarrow NO\phi$ is not valid (see Example 1).

NO At the next step you will be obliged to ensure that ...
ON You are now obliged to ensure that by the next step ...

The following formulae 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 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 axiomitisation of RoCTL*. However, it will examine which axioms of A are also valid for $\blacktriangle$ and O . The axioms [18] 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	$AG(A\phi \rightarrow EN(A\psi UA\phi)) \rightarrow (A\phi \rightarrow EG(A\psi UA\phi))$		

If $O\phi \rightarrow \phi$ or $\phi \rightarrow OP\phi$ were valid, this would imply that everything which was true was also permissible. As mentioned previously $ON\phi \rightarrow NO\phi$ is not

valid in RoCTL*. 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 $\blacktriangle G\left(A\phi \rightarrow PN\left(A\psi U A\phi\right)\right) \rightarrow \left(A\phi \rightarrow \Delta G\left(A\psi U A\phi\right)\right)$ is also valid in RoCTL*.

3.3 Validities

We will now show that the O operator behaves much like the O operator from Standard Deontic Logic (SDL). The Deontic axiom $O\phi \rightarrow P\phi$ is valid in RoCTL* and many invalid statement forms from SDL are also invalid in RoCTL*.

Lemma 1. *The axiom class $O\phi \rightarrow P\phi$ is valid in RoCTL* (and SDL).*

Proof. If $M, \sigma \vDash O\phi$, then for all failure-free fullpaths π starting with σ_0 we have $M, \pi \vDash \phi$. From the seriality of $\xrightarrow{s}$ there exists at least one failure-free fullpath starting with σ_0. Hence $P\phi$.

Note 2. The A, O and N operators of RoCTL* are normal and O satisfies the axiom class D. Hence, all statements that are valid in SDL (KD) are valid in RoCTL*. In fact RoCTL* is strongly complete on the set of formulæ of D as $O\phi \leftrightarrow \phi$ where ϕ is a state formula.

Lemma 2. *For all formulæ ϕ, the formula $\psi = (\blacktriangle N\phi \rightarrow N\blacktriangle\phi)$ is valid in RoCTL*.*

Proof. Say that there exists a RoCTL-structure M and fullpath τ such that $M, \tau \nvDash \psi$. Then $M, \tau \nvDash N\blacktriangle\phi$ and $M, \tau \vDash \blacktriangle N\phi$. From $M, \tau \vDash \blacktriangle N\phi$ we know that for all deviations σ from τ we have $M, \sigma \vDash N\phi$.

From $M, \tau \nvDash N\blacktriangle\phi$ we know that $M, \tau_{\geq 1} \nvDash \blacktriangle\phi$, and so for some deviation ρ from $\tau_{\geq 1}$ we have $M, \pi_{\geq 1} \nvDash \phi$. As ρ is a deviation from $\tau_{\geq 1}$, it must also be an i-deviation for some i. We see that if we prefix ρ with τ_0 to form a new fullpath π, we have

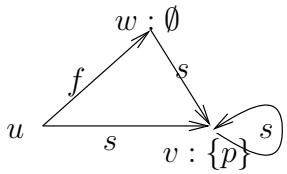
$$\pi_{\leq 1} = \tau_{\leq 1},$$

and as $\pi_{\geq 1}$ is an i-deviation from $\tau_{\geq 1}$ we know that

$$\pi_{\leq i+1} = \tau_{\leq i+1} \text{ and } \pi_{\geq i+2} \in S(\pi_{i+2}).$$

Thus, π is an $i+1$-deviation from τ. However, $M, \pi \not\models N\phi$, which contradicts the requirement that all deviations σ from τ satisfy $M, \sigma \models N\phi$. Hence, by contradiction, we may say that ψ is valid in RoCTL*.

Lemma 3. *The axiom class $(N\blacktriangle\phi \rightarrow \blacktriangle N\phi)$ is not valid in RoCTL*.*



Proof. Let $\sigma = \langle u, v, v, \ldots \rangle$. We see that $M, \sigma_{\geq 1} \models p$, thus $M, \sigma_{\geq 1} \models \blacktriangle p$ and so $M, \sigma \models N\blacktriangle p$. However, we see that $\pi = \langle u, w, v, \ldots \rangle$ is a 0-deviation from σ and $M, \pi \not\models Np$ which means that $M, \sigma \not\models \blacktriangle N\phi$.

Lemma 4. *Neither the axiom class $\phi \rightarrow P\phi$, nor the axiom class $O\phi \rightarrow \phi$, is valid in RoCTL* (or SDL)*

Proof. Say we have a RoCTL-structure $M = (A, \xrightarrow{s}, \xrightarrow{f}, \alpha)$ such that $A = \{u, v\}$, we have $\xrightarrow{s} = \{(u, v), (v, v)\}$, we have $\xrightarrow{f} = \{(u, u), (v, u)\}$, we have $\alpha(u) = \emptyset$ and $\alpha(v) = \{p\}$.

Let σ be the fullpath $\{v, u, u, u, \ldots\}$, we see that $M, \sigma \models N\neg p$ but for every fullpath π in $S(v)$, we have $M, \pi \models Np$. Hence $M, \sigma \not\models N\neg p \rightarrow PN\neg p$, and the axiom class $\phi \rightarrow P\phi$ is not valid in RoCTL*. Likewise we may show that the axiom class $O\phi \rightarrow \phi$ is not valid.

Corollary 1. *The axiom class $\phi \rightarrow O\phi$ is not valid.*

Lemma 5. *The axiom class $p \rightarrow Op$ is valid in RoCTL* (unlike SDL)*

Proof. Every fullpath in $S(\sigma_0)$ starts with σ_0. Hence if $M, \sigma \models p$ then also $M, \sigma \models Op$.

Note 3. The above lemma apparently shows a difference between the RoCTL* O and the SDL O. However, the difference may be in the variables used. SDL was founded with the assumption that you cannot have contradictory obligations, and thus $O\phi \rightarrow P\phi$. Since we want to able to separate failures out into

independent failures, we want it to be impossible for a single deviation to force the system to fail multiple times. Thus we want a strong interpretation of non-contradictory obligations, where the obligations are not only logically consistent, but also consistent with the real world. As it is impossible to change the present, by definition any rule that requires you to change the present is impossible to fulfill. Even if a failure caused the defect in the present, it would be unfair to count the inability to change the present as yet another failure. Thus we want $O\phi$ to imply that it is not too late for us ensure ϕ.

It is also easy to prove that if ϕ is a formula in SDL and ϕ' is ϕ with every variable prefixed with N (or NG) then ϕ is valid in SDL iff ϕ' is valid in RoCTL*.

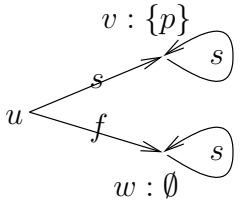
4 Examples

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

$ON(Gp)$**:** You should commit to the proper decision.
$NO\,(G\neg p \vee Gp)$**:** 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 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:

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



Say that σ is the fullpath $\langle u, w, w, \ldots\rangle$. 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 $M, \sigma \nvDash (ON\phi \to NO\phi)$ where $\phi = p$. Likewise $M, \sigma \vDash (NO\neg p \wedge \neg ON\neg p)$, so $M, \sigma \nvDash (NO\phi \to ON\phi)$ 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. We wish to develop a communication protocol such that no general will commit to an attack unless the other general also commits to an attack, while still allowing the possibility of organising a coordinated attack if enough messages get through.

$AG\,(s_A \rightarrow ONr_B)$**:** If A sends a message, B should receive it at the next step.
$AG\,(\neg s_A \rightarrow \neg Nr_B)$**:** 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.
$A\,(\neg r_B W\,(r_B \wedge (s_B \wedge Ns_B \wedge NNf_B)))$**:** Once general B receives plans, B will send two messages to A and then commit to an attack.

We expect to find that the conjunction $\hat{\phi}$ of the above formulæ are consistent under RoCTL*, and that $\hat{\phi}$ implies correct behaviour even if a single failure occurs:

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

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:

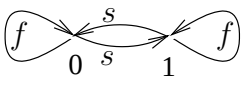
b "The cat bowl is full at the beginning of this hour"
d "This hour is feeding time"

We can translate the statements above into RoCTL* statements:

1. $AG(d \to \neg Nd)$: If this hour is feeding time, the next is not.
2. $AG((d \vee \neg b) \to \Delta N \neg b)$: 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. $AG((\neg d \wedge b) \to Nb)$: If the bowl is full and it is not feeding time, the bowl will be full at the beginning of the next hour.
4. $O\blacktriangle G\,(d \to b)$: 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.

We expect to find the formulæ above to be consistent. We also expect to be able to derive the formula $\blacktriangle GONb$, meaning that the bowl must be filled at every step (in case we forget at the next step), unless we have already failed twice. We also expect to derive the formula $AGONb \to O\blacktriangle G\,(d \to b)$, 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. Say that a bit ought to flip at every step, but might fail to flip at any particular step. This may be represented as:



$AGO\,(b \leftrightarrow \neg Nb)$

Then we may derive the following statements

$O\blacktriangle((b \wedge Nb) \to NG\,(b \leftrightarrow \neg Nb))$ If a single failure occurs, and the bit fails to flip at the next step, it will flip continuously from then on.

$O\blacktriangle FG\,(b \leftrightarrow \neg Nb)$ Even if a single failure occurs, there will be time at which the bit will flip correctly from then on.

However, we will not be able to derive $OF\blacktriangle G\,(b \leftrightarrow \neg Nb)$, as this would mean that there was a time at which a failure could not cause the bit to miss a step.

Example 5. Say that we have wireless sensor and a base station. Upon detecting some event, the wireless sensor will activate and send three packets to the base station. The base station will not know that the wireless sensor sent data if all three packets were lost. Thus an error will be reported iff the base station receives either one or two packets. This can be formalised as

$s \wedge Ns \wedge NNs \wedge NNNG\neg s$: The sensor will send three packets.

$AG\,(s \rightarrow ONr \wedge \neg s \rightarrow N\neg s)$: If a packet is sent, it should be received at the next step. If it is not sent it will not be received.

$\neg N\,((r \wedge Nr \wedge NNr) \vee G\neg r) \rightarrow NNNe$: An error will be detected if some packets, but not all three, are received.

It follows that $O\blacktriangle\,(\Delta FeU\neg s)$, indicating that it is robustly true that if an additional failure occurs, an error could be detected. In this example a failure may not indicate a packet being dropped, e.g. it has not been specified whether the packet arrives corrupted. Thus the system cannot detect all failures. In RoCTL* it is impossible to specify that a failure will have an effect. At best we can specify that it is always *possible* for a failure to be detected. However, we can specify that some particular effect will be detected. For example, we can express the statement "Even if two or fewer packets are lost, either all packets arrive or an error is detected" as

$$O\blacktriangle\blacktriangle N\,((r \wedge Nr \wedge NNr) \vee Fe)\ .$$

Example 6. Say a system has a battery that can sustain the system for a single step, even if a failure occurs (the fuse blows). Let ϕ represent "the system has power now and at the next step". Then, even if a single failure occurs, it will always be the case that even if a deviating event occurs the system will have power now and at the next step ($OG\blacktriangle\phi$). It would not follow that even if a single failure occurred the system would always have power ($O\blacktriangle G\phi$); the battery power would only last one step after the fuse blew. If we also specified that the fuse was an electronic fuse that automatically reset, then if a single failure occurs, the system would only have to rely on battery power for one step. Then, if the fuse only blows once then system will always have power ($\blacktriangle G\phi$).

As with the A operator in CTL*, $\blacktriangle G\phi \rightarrow G\blacktriangle\phi$ is valid in RoCTL* but $G\blacktriangle\phi \rightarrow \blacktriangle G\phi$ is not.

5 Embeddings

Lemma 6. *CTL* is embedded in RoCTL*.*

Proof. For any RoCTL-structure $M = (A, \xrightarrow{s}, \xrightarrow{f}, \alpha)$, we say the structure $C = \left(A, \xrightarrow{sf}, \alpha\right)$ is the CTL-equivalent to M. As $\xrightarrow{s}$ is serial, $\xrightarrow{sf}$ is serial too, and so C is a CTL-structure. We see that a sequence of worlds is a fullpath through M iff

it is a fullpath through C. Recall that the function $\mathcal{SF}(a)$ was defined as the set of all fullpaths that start at a.

The operators from classical logic behave the same in all normal logics such as RoCTL* and CTL*. Recall that the semantic definition of N, U and A were defined as follows:

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

We see that the definitions above are the same as in CTL*, and so a CTL* formula is valid on a RoCTL-structure iff it is valid on the CTL-equivalent CTL-structure. It is clear that every CTL-structure has a RoCTL-structure that is CTL-equivalent to it. Hence a CTL* formula is valid in RoCTL* iff it is valid in CTL.

Lemma 7. *Say 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 of CTL*.*

Say $M = (A, \xrightarrow{s}, \xrightarrow{f}, \alpha)$ is a RoCTL-structure. Recall that the function $S(a)$ was defined as the set of failure-free fullpaths that start at a, and that a failure-free fullpath is a fullpath that traverses only $\xrightarrow{s}$, whereas $\mathcal{SF}(a)$ was defined as the set of all fullpaths that start at a. Note that in RoCTL* $M, \sigma \vDash O\psi'$ iff for all paths π in $S(\sigma_0)$, we have $M, \sigma \vDash \psi'$. Hence it is the case that $O\psi'$ is a RoCTL* validity iff it is the case that for all RoCTL-structures M, and for all failure-free paths σ we have $M, \sigma \vDash \psi'$. Recall that the semantic definition of O was as follows:

$$M, \sigma \vDash O\phi \text{ iff } \forall_{\pi \in S(\sigma_0)} M, \pi \vDash \phi$$

We see that the definition of O above is the same as the definition of A except with $\mathcal{SF}$ replaced with S in CTL*. As with $\xrightarrow{sf}$, the only requirement on $\xrightarrow{s}$ is that it is serial. As with Lemma 6 above, we see that $O\psi'$ is a valid formula of RoCTL* iff ψ is a valid formula of CTL*, Hence, from Lemma 6 we see that $O\psi'$ is a validity of RoCTL* iff ψ is a validity of RoCTL*.

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

$$w_i^s \to w_j^s \iff w_i \xrightarrow{s} w_j \qquad w_i^s \to w_j^f \iff w_i \xrightarrow{f} w_j$$
$$w_i^f \to w_j^s \iff w_i \xrightarrow{s} w_j \qquad w_i^f \to w_j^f \iff w_i \xrightarrow{f} w_j$$

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 $M^* = (A^*, \to^*, \alpha^*)$ 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 worlds w_i and w_j we have:

$$w_i \xrightarrow{s} w_j \iff (w_i \to^* w_j) \wedge (\text{Viol} \notin \alpha(w_j))$$
$$w_i \xrightarrow{f} w_j \iff (w_i \to^* w_j) \wedge (\text{Viol} \in \alpha(w_j))\,.$$

5.2 Translating a RoCTL* formula into a QCTL* formula

Definition 8. *Given some CTL structure $M = (A, \to, \alpha)$ and some $x \in A$, an <u>x-variant</u> of M is some structure $M = (A, \to, \alpha')$ 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

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

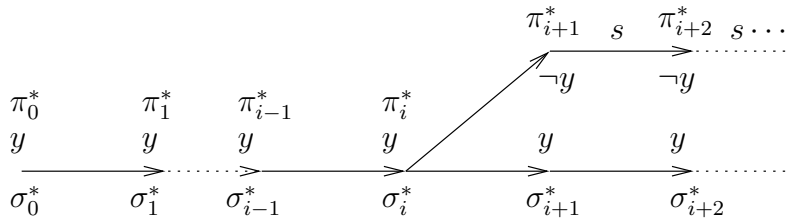
This paper uses the tree semantics for QCTL*. These semantics are the same as the Kripke semantics except that, whereas the Kripke semantics evaluates validity over the class $\mathbb{C}$ of CTL-structures, the tree semantics evaluate validity over the class $\mathbb{C}_t$ of tree CTL-structures. This changes the validities of the logic as, unlike CTL* [7], QCTL* is sensitive to unwinding into a tree structure [15].

We let γ be the (Q)CTL* formula $NNG\neg\text{Viol}$. Thus γ does not specify whether the previous or next transitions are failures, but requires that all transitions after the next one be successes. 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 function ϕ^* and for some atom y not in ϕ^*:

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

Note that for $t^{\Delta}(\phi^*)$ to hold, $E[(Gy \vee F(y \wedge \gamma)) \wedge \phi^*]$ must hold for all possible variables 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 8. *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 $M, \sigma \vDash \phi$ iff $M^*, \sigma^* \vDash \phi^*$. Then, for all M and σ it is the case that $M, \sigma \vDash \Delta\phi$ iff $M^*, \sigma^* \vDash t^{\Delta}(\phi^*)$.*

Proof. ($\Longrightarrow$) Say that $M, \sigma \vDash \Delta\phi$. Then $M, \sigma \vDash \phi$ or there exists a deviation π from σ such that $M, \pi \vDash \phi$. If $M, \sigma \vDash \phi$ then $M^*, \sigma^* \vDash \phi^*$ and so

$$M^*, \sigma^* \vDash \forall_y [Gy \rightarrow E[Gy \wedge \phi^*]],$$

thus $M^*, \sigma^* \vDash t^{\Delta}(\phi^*)$.

If $M, \sigma \nvDash \phi$ then, for some i, there exists an i-deviation π from σ such that $M, \pi \vDash \phi$. 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 $M^*, \pi^*_{\geq i} \vDash \gamma$ and $M^*, \pi^* \vDash F(y \wedge \gamma) \wedge \phi^*$ from which it follows that $M^*, \sigma^* \vDash t^{\Delta}(\phi^*)$.

($\Longleftarrow$) 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 $M^y, \sigma^* \vDash E\left[(Gy \vee F\left(y \wedge \gamma\right)) \wedge \phi^*\right]$. Thus there exists some fullpath σ^y such that $\sigma^y_0 = \sigma^*_0$ and $M^y, \sigma^y \vDash F\left(y \wedge \gamma\right) \wedge \phi^*$ 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 $M^y, \sigma^y \vDash F\left(y \wedge \gamma\right) \wedge \phi^*$ then there exists a non-negative 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 the translation function $t^{\Delta}\left(\phi^*\right)$ defined above, together with $t^{O}(\phi^*) \equiv (NG\neg\text{Viol} \rightarrow \phi^*)$, $t^{\neg}(\phi^*) \equiv \neg\phi^*$, $t^{N}(\phi^*) \equiv N\phi^*$, $t^{A}(\phi^*) \equiv A\phi^*$, $t^{\wedge}(\phi^*_1, \phi^*_2) \equiv \phi^*_1 \wedge \phi^*_2$, $t^{U}(\phi^*_1, \phi^*_2) \equiv \phi^*_1 U \phi^*_2$, $t^{\top} = \top$ 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 2. *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 [9,11], it is obvious from Theorem 1 that RoCTL* is decidable.

6 Conclusion

We have proposed a logic for dealing with multiple failures. 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 an elementary decision procedure [20,10].

Finding an Axiomitisation for RoCTL* may be a challenging task. Although this paper has compared the validities of the ▲ and O operators to the axioms for the A operator, we are far from finding a sound axiomitisation of RoCTL*, and have not examined how to prove that such an axiomitisation 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. Huib Aldewereld, Davide Grossi, Javier Vazquez-Salceda, and Frank Dignum. Designing normative behaviour by the use of landmarks. In *Agents, Norms and Institutions for Regulated Multiagent Systems*, Utrecht, The Netherlands, July 2005.
2. A. Arnold, A. Vincent, and I. Walukiewicz. Games for synthesis of controllers with partial observation. *Theor. Comput. Sci.*, 303(1):7–34, 2003.
3. Nuel Belnap. Backwards and forwards in the modal logic of agency. *Philosophy and Phenomenological Research*, 51(4):777–807, Dec 1991.
4. Jan Broersen, Frank Dignum, Virginia Dignum, and John-Jules Ch. Meyer. *Designing a Deontic Logic of Deadlines*, volume 3065/2004 of *Lecture Notes in Computer Science*, pages 43–56. Springer, 2004. doi: 10.1007/b98159.
5. E. Clarke and E. Emerson. Synthesis of synchronization skeletons for branching time temporal logic. In *Proc. IBM Workshop on Logic of Programs, Yorktown Heights, NY*, pages 52–71. Springer, Berlin, 1981.
6. M. de Weerdt, A. Bos, H. Tonino, and C. Witteveen. A resource logic for multi-agent plan merging, 2001.
7. E. Emerson. Alternative semantics for temporal logics. *TCS*, 26, 1983.
8. E. A Emerson and A. P Sistla. Deciding full branching time logic. Technical report, University of Texas at Austin, Austin, TX, USA, 1985.

9. E. Allen Emerson and A. Prasad Sistla. Deciding branching time logic. In *STOC '84: Proceedings of the sixteenth annual ACM symposium on Theory of computing*, pages 14–24, New York, NY, USA, 1984. ACM Press.
10. T. French. *Bisimulation Quantifiers for Modal Logics.* PhD thesis, University of Western Australia, 2006.
11. Tim French. Decidability of quantifed propositional branching time logics. In *AI '01: Proceedings of the 14th Australian Joint Conference on Artificial Intelligence*, pages 165–176, London, UK, 2001. Springer-Verlag.
12. Tim French, John C. M^{c}Cabe-Dansted, and Mark Reynolds. A temporal logic of robustness, RoCTL*. Technical report, 2007. `dansted.org/papers/RoCTL07.pdf`.
13. Hans Hansson and Bengt Jonsson. A logic for reasoning about time and reliability. *Formal Aspects of Computing*, 6(5):512–535, 1994.
14. Thierry Jéron, Hervé Marchand, Sophie Pinchinat, and Marie-Odile Cordier. Supervision patterns in discrete event systems diagnosis. In *8th International Workshop on Discrete Event Systems*, pages 262–268, July 2006.
15. O. Kupferman. Augmenting branching temporal logics with existential quantification over atomic propositions. In *Computer Aided Verification, Proc. 7th Int. Conference*, pages 325–338, Liege, 1995. Springer-Verlag.
16. W. Long, Y. Sato, and M. Horigome. Quantification of sequential failure logic for fault tree analysis. *Reliability Engineering and System Safety*, 67:269–274, 2000.
17. L. Thorne McCarty. Defeasible deontic reasoning. *Fundam. Inform.*, 21(1/2):125–148, 1994.
18. Mark Reynolds. An axiomatization of full computation tree logic. *The Journal of Symbolic Logic*, 66(3):1011–1057, September 2001.
19. AGERRI Rodrigo and ALONSO Eduardo. *Normative pragmatics for agent communication languages*, volume 172-181, pages 172–181. Springer, 2005.
20. A. Prasad Sistla, M. Y. Vardi, and P. Wolper. The complementation problem for buchi automata with applications to temporal logic. *Theor. Comput. Sci.*, 49(2-3):217–237, 1987.
21. Leendert W. N. van der Torre and Yaohua Tan. The temporal analysis of Chisholm's paradox. In Ted Senator and Bruce Buchanan, editors, *Proceedings of the Fourteenth National Conference on Artificial Intelligence and the Ninth Innovative Applications of Artificial Intelligence Conference*, pages 650–655, Menlo Park, California, 1998. AAAI Press.