Counting CTL
François Laroussinie, Antoine Meyer, Eudes Péttonnet

To cite this version:
François Laroussinie, Antoine Meyer, Eudes Péttonnet. Counting CTL. FoSSaCS 2010, Mar 2010, Paphos, Cyprus. pp.206-220, 10.1007/978-3-642-12032-9_15. hal-00681263

HAL Id: hal-00681263
https://hal.science/hal-00681263
Submitted on 21 Mar 2012

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L’archive ouverte pluridisciplinaire HAL, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d’enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.
Counting CTL

François Laroussinie\textsuperscript{1*}, Antoine Meyer\textsuperscript{2}, and Eudes Petonnet\textsuperscript{1}

\textsuperscript{1} LIAFA, Université Paris Diderot – Paris 7 & CNRS UMR 7089, France
\{Francois.Laroussinie,Eudes.Petonnet\}@liafa.jussieu.fr
\textsuperscript{2} LIGM, Université Paris Est – Marne-la-Valle & CNRS UMR 8049, France
Antoine.Meyer@univ-mlv.fr

Abstract. This paper presents a range of quantitative extensions for the temporal logic CTL. We enhance temporal modalities with the ability to constrain the number of states satisfying certain sub-formulas along paths. By selecting the combinations of Boolean and arithmetic operations allowed in constraints, one obtains several distinct logics generalizing CTL. We provide a thorough analysis of their expressiveness and of the complexity of their model-checking problem (ranging from P-complete to undecidable).

1 Introduction

Among the existing approaches to the formal verification of automated systems, model checking \cite{7,17} aims at automatically establishing the validity of a certain formal specification (modeled as a formula in a suitable logic) over the system under study (modeled for instance as a finite automaton). This set of techniques is now well-established and successful, with many real-world applications.

To formalize the specification of temporal properties, for instance in the case of reactive systems, temporal logics (TL) were proposed thirty years ago \cite{16} and widely studied since. They are today used in many model-checking tools. There exists a wide variety of temporal logics, differing for instance by the models over which formulas are interpreted or by the kind of available temporal modalities.

Two well-known examples are LTL in the linear-time framework (where formulas are interpreted over infinite runs) and CTL for the branching-time case (where formulas are interpreted over states of Kripke structures). See \cite{8} for a survey of classical temporal logics for systems specification.

Temporal logics have been extended in various ways in order to increase their expressive power. For example, while LTL and CTL only contain future operators, it is also possible to consider past-time modalities to express properties of the past of a run. One can also extend temporal logics with regular expressions (see for instance \cite{19,10}). Other extensions were proposed to handle quantitative aspects of systems. For example, some logics can contain timing constraints to specify that some event $P_1$ has to occur less than 10 time units before another event $P_2$. Such temporal logics, like TCTL \cite{2,9}, have been especially studied in

\* Partly supported by ANR project DOTS and project QUASIMODO (FP7-ICT).
the framework of timed model checking. Another quantitative extension consists in probabilistic logics where one can specify probability bounds over the truth of some property (see for example [4]).

We propose several extensions of CTL with constraints over the number of states satisfying certain sub-formulas along runs. For example, considering a model for an ATM, we can express the property “whenever the PIN is locked, at least three erroneous attempts have been made” by: $\neg \text{EF}_{[\#\text{error} \leq 2]} \text{lock}$ (one cannot reach a state where the PIN is locked but less than two errors have occurred). Similarly, $\neg \text{EF}_{[\#\text{error} \geq 3]} \text{money}$ states that three mistakes forbid cash retrieval. We use subscripts on the temporal modality (as in TCTL) to associate a constraint with the run for which the modality holds. Note that these two properties could be clearly stated in CTL by nesting $\exists \cup$ modalities, but the resulting formulas would probably be too big to be easily handled by the user of a model checker. For each extension we consider, we study its expressiveness compared to CTL. In some cases, there is no formal gain of expressiveness because there exist natural translations to obtain equivalent CTL formulas, but these extensions are often exponentially more succinct than CTL: they allow writing concise specifications that would require formulas of exponentially larger size in CTL. In other cases, we show that adding some constraints increases the expressive power of CTL.

We consider the model checking problem for various sets $C$ of constraints, and denote by $CCTL_C$ the corresponding extension of CTL. We show that polynomial-time algorithms exist when considering Until modalities with constraints of the form $\sum_i \#\phi_i \sim c$ with $\sim \in \{<,\leq,=,\geq,>\}$ and $c \in \mathbb{N}$. Additionally allowing Boolean combinations of such constraints or integer coefficients in the sum (or both) makes model checking $\Delta^P_2$-complete. We also consider the case of diagonal constraints $\#\phi - \#\psi \sim c$ and their more general form $\sum_i \pm \#\phi_i \sim c$ with $c \in \mathbb{Z}$ and show that model checking can be done in polynomial time. However, allowing Boolean combinations of such constraints leads to undecidability. Finally we define a version of $CCTL$ with freeze variables and show that it induces a complexity blow-up: model checking becomes PSPACE-complete.

Several existing works provide related results. In [10], an extension of LTL with a kind of regular expressions containing quantitative constraints over the number of occurrences of sub-expressions is presented. This extension yields algorithms whose time complexity is exponential in the size of formulas and the value of integer constants. In [11], extensions of CTL are defined including parameters in constraints. One of these formalisms, namely GPCTL, allows one to express properties with constraints defined as positive Boolean combinations of sums of the form $\sum_i P_i \leq c$ where every $P_i$ is an atomic proposition. Model-checking $\exists \cup$ formulas with such a constraint is shown to be NP-complete and a polynomial algorithm is given for a restricted logic (with parameters). In [20], a branching-time temporal logic with general counting constraints (using a variant of freeze variables) is defined to specify event-driven real-time systems. To

\footnote{For complexity results, we always assume that integer constants are encoded in binary.}
obtain decidability, they restrict the analysis to systems verifying some bounded
progress condition. In [6, 5], extensions of LTL and CTL with Presburger con-
straints over the number of states satisfying some formulas are considered, for
some class of infinite state processes. The complexity of these problems is much
higher than the cases we are concerned with. Finally there also exist timed ex-
tensions of CTL interpreted over Kripke structures (see for instance [9]).

The paper is organized as follows. In Section 2, we introduce the definitions
of the main formalisms we will use. In Section 3, we show that several of our
proposed logics are not more expressive than the classical CTL, yet exponentially
more succinct. In Section 4, we address the model-checking problem and provide
exact complexity results for most of the logics we introduce. Finally we present
in Section 5 a different logic with freeze variables, together with the complexity
of its model-checking problem.

2 Definitions

2.1 Models

Let AP be a set of atomic propositions. In branching-time temporal logics, for-
mulas are interpreted over states of Kripke structures.

Definition 1. A Kripke structure (or KS) S is a tuple ⟨Q, R, ℓ⟩ where Q is a
finite set of states, R ⊆ Q × Q is a total accessibility relation and ℓ : Q → 2AP
is a labelling of states with atomic propositions.

A run ρ of S is an infinite sequence of states q0q1q2 . . . such that (qi, qi+1) ∈ R
for every i. We use ρ(i) to denote the state qi and ρi to denote the prefix q0 · · · qi
of ρ. Runs(q) denotes the set of runs starting from some state q ∈ Q. We write
σ ≤ ρ when σ is a prefix of ρ.

We will also consider Durational Kripke Structures (DKS), where an integer
duration is associated with every transition. Thus for a DKS S = ⟨Q, R, ℓ⟩, we
have R ⊆ Q × N × Q. The duration of a transition is also called a weight or a
cost. We use DKS0/1 to denote the DKSs where the durations belong to {0, 1}.
The notion of duration is naturally extended to finite runs of DKSs.

2.2 Counting CTL

We define several extensions of CTL able to express constraints over the number
of times certain sub-formulas are satisfied along a run.

Definition 2. Given a set of atomic propositions AP and a set of constraints
C, we define:

\[
\begin{align*}
\text{CCTL}_C & \ni \varphi, \psi ::= P \mid \varphi \land \psi \mid \neg \varphi \mid E\varphi U_{[C]} \psi \mid A\varphi U_{[C]} \psi
\end{align*}
\]

4 By total relation, we mean a relation R ⊆ Q × Q such that ∀p ∈ Q, ∃q ∈ Q, (p, q) ∈ R.
where \( P \in \text{AP} \) and \( C \in \mathcal{C} \). The sets of constraints we consider are defined as follows, with \( \ell, k, l, k' \in \mathbb{N}, k' \in \mathbb{Z} \) and \( \sim \in \{<, \leq, =, \geq, >\} \). First we have the sets of atomic constraints:

\[
\begin{align*}
\mathcal{C}_0 \ni C & \equiv \sharp \varphi \sim k & \text{with } \varphi \in \text{CCTL}_{\mathcal{C}_0} \\
\mathcal{C}_1 \ni C & \equiv (\Sigma_{i=1}^k \varphi_i) \sim k & \text{with } \varphi_i \in \text{CCTL}_{\mathcal{C}_1} \\
\alpha \mathcal{C}_1 \ni C & \equiv (\Sigma_{i=1}^k \alpha_i \cdot \sharp \varphi_i) \sim k & \text{with } \varphi_i \in \text{CCTL}_{\alpha \mathcal{C}_1}, \alpha_i \in \mathbb{N} \\
\mathcal{C}_2 \ni C & \equiv (\sharp \varphi - \sharp \psi) \sim k' & \text{with } \varphi, \psi \in \text{CCTL}_{\mathcal{C}_2} \\
\mathcal{C}_3 \ni C & \equiv (\Sigma_{i=1}^k \pm \varphi_i) \sim k' & \text{with } \varphi_i \in \text{CCTL}_{\mathcal{C}_3} \\
\alpha \mathcal{C}_3 \ni C & \equiv (\Sigma_{i=1}^k \alpha_i \cdot \pm \varphi_i) \sim k' & \text{with } \varphi_i \in \text{CCTL}_{\alpha \mathcal{C}_3}, \alpha_i \in \mathbb{Z}
\end{align*}
\]

Let \( \mathcal{L}_a \) be the set \( \{\mathcal{C}_0, \mathcal{C}_1, \alpha \mathcal{C}_1, \mathcal{C}_2, \mathcal{C}_3, \alpha \mathcal{C}_3\} \). We also consider the set of constraints \( \mathcal{B}(C) \) for every \( C \in \mathcal{L}_a \), defined as the set of Boolean combinations of atomic constraints in \( C \) with sub-formulas in \( \text{CCTL}_B(C) \). We use \( \mathcal{L}_b \) for \( \{\mathcal{B}(C) | C \in \mathcal{L}_a\} \).

Finally \( \mathcal{L}_{\text{cons}} \equiv \mathcal{L}_a \cup \mathcal{L}_b \).

We make use of the standard abbreviations \( \lor, \rightarrow, \iff, \bot, \top \), as well as the additional modalities \( \text{EF}_{C} \varphi \equiv \text{ETU}_{C} \varphi, \text{AF}_{C} \varphi \equiv \text{ATU}_{C} \varphi \), and their duals \( \text{AG}_{C} \varphi \equiv \neg \text{EF}_{C} \neg \varphi \) and \( \text{EG}_{C} \varphi \equiv \neg \text{AF}_{C} \neg \varphi \). Any formula occurring in a constraint \( C \) associated with a modality in \( \Phi \) is considered as a sub-formula of \( \Phi \). The size \( |\Phi| \) of \( \Phi \) takes the size of these constraints and their sub-formulas into account, assuming that integer constants are encoded in binary (unless explicitly stated otherwise). The DAG-size of \( \Phi \) is the total number of distinct sub-formulas of \( \Phi \). As model-checking algorithms compute only once the truth value of a sub-formula, this is generally more relevant to the complexity of model-checking.

The semantics of \( \text{CCTL}_C \) formulas (with \( C \in \mathcal{L}_{\text{cons}} \)) is defined over Kripke structures as follows:

**Definition 3.** The following clauses define the conditions for a state \( q \) of some KS\( S = (Q, R, l) \) to satisfy a \( \text{CCTL}_C \) formula \( \varphi \) — written \( q \models_S \varphi \) — by induction over the structure of \( \varphi \) (we omit Boolean modalities):

\[
q \models_S E\varphi_R U_C \psi \iff \exists \rho \in \text{Runs}(q), \exists i \geq 0, \rho(i) \models_S \psi, \rho_{i-1} \models_S C, \text{ and } \forall 0 \leq j < i, \rho(j) \not\models_S \varphi
\]

\[
q \models_S A\varphi_R U_C \psi \iff \forall \rho \in \text{Runs}(q), \exists i \geq 0, \rho(i) \models_S \psi, \rho_{i-1} \models_S C, \text{ and } \forall 0 \leq j < i, \rho(j) \not\models_S \varphi
\]

Let \( C \in \mathcal{L}_{\text{cons}} \) be a set of constraints and \( C \) be a constraint in \( C \), the semantics of \( \rho_i \models_S C \) is based on the interpretation of \( \sharp \varphi \) over \( \rho_i \), denoted by \( \models_{\rho_i} \varphi \) and defined as: \( \models_{\rho_i} \varphi \equiv \{ j \mid 0 \leq j \leq i \land \rho(j) \models_S \varphi \} \). Given these values, \( C \) is interpreted in a natural way.

In the following we omit the subscript \( S \) for \( \models \) when no confusion is possible. We use \( \equiv \) to denote the standard equivalence between formulas.

**Remark 1.** The classical X operator ("neXt") can be expressed in \( \text{CCTL}_{\mathcal{C}_0} \) as \( \text{EX} \varphi \equiv \text{EF}_{[\text{T}=1]} \varphi \), and that basic constraints in \( \mathcal{C}_0 \) can be expressed in \( \mathcal{C}_2 \) because
\[ \varphi \equiv \neg \varphi \rightarrow \bot. \] Moreover, for CCTL\(_{B(C)}\) with \( C \in L_\alpha \), since \( \varphi U_C \psi \equiv F_{[C \land \neg \varphi = 0]} \psi \), the modality \( F \) is sufficient to define \( U \); thus such a logic CCTL\(_{B(C)}\) can also be built from atomic propositions using Boolean operators and modalities \( EF_{[C]} \varphi \) and \( AF_{[C]} \varphi \) (or \( EG_{[C]} \varphi \)). Note that all these translations are succinct (linear in the size of formulas) and do not have any impact on complexity results.

**Remark 2.** The related temporal logic TCTL [2], whose semantics are defined over *timed* models, allows one to label temporal modalities with duration constraints. For instance, one may write \( \varphi \Uparrow \leq k \psi \) to express the fact that \( \varphi \) is consistently true until, before \( k \) time units have elapsed. \( \psi \) eventually holds. When all transitions in a durational Kripke structure have duration 1 (i.e. the duration of any run is equal to its length), TCTL (or RTCTL in [9]) formulas can be directly coded into the logic CCTL\(_C\) by only using the sub-formula \( \top \) inside constraints. A similar coding is also possible when one uses a proposition \( \text{tick} \) to mark time elapsing as in [15].

### 2.3 Examples of CCTL formulas.

Consider a model for an ATM, whose atomic propositions include \( \text{money} \), \( \text{reset} \) and \( \text{error} \), with the obvious meaning. To specify that it is not possible to get money when three mistakes are made in the same session (i.e. with no intermediate reset), we can use the formula \( AG(\neg EF[\text{error} \geq 3 \land \text{reset} = 0] \text{money}) \) that belongs to CCTL\(_{B(C)}\). Note that this could also be expressed by the CCTL\(_C\) formula \( AG(\neg E(\neg \text{reset}) U [\text{error} \geq 3] \text{money}) \).

Consider a mutual exclusion algorithm with \( n \) processes trying to reach their critical section. We can specify that it verifies the bounded waiting property with bound 10 (i.e. when a process \( P \) tries to reach its CS, then at most 10 other processes can reach theirs before \( P \) does) by the following CCTL\(_{B(C)}\) formula: 
\[
AG \bigwedge_i \left( \text{request}_i \Rightarrow \neg EF[\sum_j \text{cs}_j > 10 \land \text{cs}_i = 0] \top \right). 
\]

CCTL\(_{C_2}\) states that along any finite run, the number of \( \text{receive} \) events cannot exceed the number of \( \text{send} \) events.

Quantitative constraints can also be useful for fairness properties. For example \( AG AF \bigwedge_i [5 \leq t_i \leq 10] \top \) expresses that the \( \varphi_i \)'s occur infinitely often along every run (as stated with the CTL formula \( \bigwedge_i (AG AF \varphi_i) \)) but it also ensures some constraint on the number of states satisfying the \( \varphi_i \)'s along every execution: for example, it is not possible to have a sub-run where \( \varphi_1 \) holds for 11 states and \( \varphi_2 \) holds only for 4 states.

Note that with CCTL\(_{C_3}\) one can express properties over the ratio of two kinds of states along a run. For example, \( EF[100 \land \text{error} \rightarrow 1 \land t < 0] P \) is true when there is a path leading to \( P \) such that the rate of \( \text{error} \) states is less than 1 percent.

Thus constraints of the form \( \frac{\#P}{\#C} \sim k \) can easily be expressed with this logic.

Finally note that we can use any temporal formula inside a constraint (and not only atomic propositions). For example, \( AG(\text{EF}[\text{time (alarm)} \leq 5] \text{init}) \) states that it is always possible to reach \( \text{init} \) with a path along which at most 5 states have a successor satisfying \( \text{alarm} \).
These examples illustrate the ability of CCTL formulas to state properties over the portion of a run leading to some state. A similar kind of properties could also be expressed with past-time modalities (like $S$ or $F^{-1}$), but unlike these modalities our constraints cannot easily describe the ordering of events in the past: they "only" allow to count the number of occurrences of formulas. We will see in the next sections that our extensions do not always induce a complexity blow-up, while model-checking $\text{CTL} + F^{-1}$ is known to be $\text{PSPACE}$-complete [14].

3 CCTL expressiveness and succinctness

When comparing two logics, the first question which comes to mind is the range of properties they can be used to define, in other words their expressiveness. When they turn out to be equally expressive, a natural way to distinguish them is then to ask how concisely each logic can express a given property. This is referred to as succinctness, and is also relevant when studying the complexity of model-checking for instance, since it may considerably influence the size of a formula required to express a given property, and hence the time required to model-check it. In this section we study the expressiveness of the different logics defined in the previous section, and provide partial results and comments about their respective succinctness with respect to $\text{CTL}$.

3.1 Expressiveness

We first show that only allowing boolean combinations and positive sums in constraints does not allow CCTL to express more properties than CTL.

Proposition 4. Any $\text{CCTL}_{B\langle \alpha C \rangle}$ formula can be translated into $\text{CTL}$.

Proof (sketch). Let $\Phi$ be a $\text{CCTL}_{B\langle \alpha C \rangle}$ formula of the form $EF\lbrack C \rbrack \varphi$ whose constraint $C$ contains $n$ counting arguments $\sharp \varphi_1$ to $\sharp \varphi_n$, each preceded by a multiplicative constant $\alpha_i$, and $m$ atomic constraints. We inductively translate $\Phi$ to CTL by building a family of formulas whose intended meaning (up to technical details) is as follows:

- If constraint $C$ holds with $\sharp \varphi_1 = 0$ for all $i$, then $\varphi$ may be true immediately.
- Otherwise, successively check for every $i$ which of the $\varphi_i$ hold in the current state, updating constraint $C$ along the way by decreasing by $\alpha_i$ the constant to which $\varphi_i$ is compared in $\Phi$.
- Once all $\varphi_i$ have been scanned, proceed to the next state and reevaluate $C$ for the new values of the constants.

Each of these steps corresponds to a sub-formula of the form $\Phi_{C',i,b}$ in the CTL translation of $\Phi$, where $C'$ is the current constraint to be checked, $i$ is the index of the formula $\varphi_i$ being scanned, and $b$ is a boolean flag used to enforce termination, $\Phi_{C,0,\bot}$ being the translation of $\Phi$ itself. By counting the number of distinct $\Phi_{C',i,b}$, one can show that the DAG-size of $\Phi_{C,0,\bot}$ is $O(n.k^m)$, where $k$ is the maximal constant appearing in $\Phi$. A similar argument holds for formulas of the form $EG\lbrack C \rbrack \varphi$ with the same resulting DAG-size. $\square$
Note that the upper bound for the above translation is parametric, and can be interpreted for all variants of CCTL below CCTL_{B \otimes C}. An example of this translation on a CCTL_{B \otimes C_1} formula is given in the next section. In contrast to this result, introducing subtractions in constraints yields a strict increase in expressiveness.

**Proposition 5.** The CCTL_{C_2} formula $\varphi = AG_{\#A-\#B<0} \perp$ cannot be translated into CTL.

*Proof (sketch).* Formula $\varphi$ (already seen in Sec. 2 with different atomic propositions) states that the number of $B$-labeled states cannot exceed the number of $A$-labeled states along any path. As shown by [3] and also presented in [18], the set of models of any CTL formula can be recognized by a finite alternating tree automaton. From such an automaton, one can easily build a finite alternating automaton over words, whose accepted language is the set of all prefixes of branches in the models of the formula, seen as words over $2^{\mathcal{AP}}$.

Suppose there exists a CTL formula $\varphi'$ equivalent to $\varphi$, and let $\mathcal{A}$ be the alternating tree automaton accepting its set of models. As stated above, from $\mathcal{A}$ one can derive a finite alternating automaton recognizing the set of all words over $2^{\{A,B\}}$ labeling a finite prefix of a branch in a model of $\varphi$, namely words whose prefixes contain at most as many $B$’s as $A$’s. Since this language is clearly not regular, this leads to a contradiction. $\Box$

3.2 Succinctness

Our extensions of CTL come with three main sources of possible concision, which appear to be orthogonal: the encoding of constants in binary, the possibility to use boolean combinations in the constraints, and the use of sums. However, only the first two prove out to yield an exponential improvement in succinctness:

**Proposition 6.** For every formula $\Phi \in CCTL_{C_1}$ with unary encoding of integers, there exists an equivalent CTL formula of DAG-size polynomial in $|\Phi|$.

This proposition is a direct consequence of Prop. 4 where $m$, the number of atomic constraints, is set to 1. For instance, to translate $\Phi = EF_{\sum_{i=1}^n p_i = K}\varphi$, we define $\forall 0 \leq k \leq K$ the family of CTL formulas:

\[
\begin{align*}
\forall 1 \leq i \leq n, 0 \leq j < n, & \quad \Phi_{i,j,k} = (p_i \land \Phi_{i+1,j+1,k}) \lor (\neg p_i \land \Phi_{i+1,j,k}) \\
\forall 1 \leq j \leq k, & \quad \Phi_{n+1,j,k} = \text{EXE}(\neg \bigvee_{i=1}^n p_i)U((\bigvee_{i=1}^n p_i) \land \Phi_{1,0,k-j}) \\
\forall j > k, & \quad \Phi_{n+1,j,k} = \bot \\
\end{align*}
\]

By construction, we have $\Phi \equiv \Phi_{1,0,K}$. The size of this family is $O(nk)$, thus the DAG-size of $\Phi_{1,0,K}$ is also polynomial in $|\Phi|$, even if its literal size is exponential. This example relies on the fact that constants are encoded in unary, to measure the impact of the addition operation in constraints. We now look at the succinctness gap due to the binary encoding of constants.
Proposition 7. $\text{CCTL}_{C_0}$ can be exponentially more succinct than $\text{CTL}$. 

Proof. In [15], it is shown that the logic $\text{TCTL}$, when interpreted over Kripke structures with a special atomic proposition $\text{tick}$ used to mark the elapsing of time, can be exponentially more succinct than $\text{CTL}$. More precisely, the $\text{TCTL}$ formulas $\text{EF}_{<n}A$ and $\text{EF}_{>n}A$, which are of size $O(\log(n))$ since $n$ is encoded in binary, do not admit any equivalent $\text{CTL}$ formula of temporal height (and hence also size) less than $n$. These formulas express the existence of a path where $A$ eventually holds and less (resp. more) than $n$ clock ticks are seen until then. They are clearly equivalent to the $O(\log(n))$-size $\text{CCTL}_{C_0}$ formulas $\text{EF}[\text{C} \sim n]A$ and $\text{EF}[\text{tick} > n]A$ respectively. □

This exhibits a first aspect in which $\text{CCTL}$ logics can be exponentially more succinct than $\text{CTL}$. However, as expressed in the next proposition, another orthogonal feature of the logic may yield a similar blow-up.

Proposition 8. $\text{CCTL}_{B(C_0)}$ with unary encoding of integers can be exponentially more succinct than $\text{CTL}$. 

Proof. It was shown by [18, 1] that any $\text{CTL}$ formula $\phi$ equivalent to the $\text{CTL}^+$ formula $\psi = \text{EF}(F P_1 \land \ldots \land F P_n)$ must be of length exponential in $n$. It turns out $\psi$ is equivalent to the $\text{CCTL}_{B(C_0)}$ formula $\psi' = \text{EF}[A_i \land P_i \geq 1] \top$, which entails the result. Note that $\psi'$ only contains the constant 1, which means that this gap cannot be imputed to the binary encoding. □

The intuitive reason for this blow-up is that a $\text{CTL}$ formula expressing the property that atomic propositions $P_1$ to $P_n$ are each seen at least once along a path would have to keep track of all possible interleavings of occurrences of $P_i$'s.

To summarize, we showed that two different aspects of the extensions of $\text{CTL}$ presented in this paper, while not increasing the overall expressiveness of the logic, may yield exponential improvements in succinctness We still have to study similar succinctness properties of the remaining $\text{CCTL}$ fragments with respect to $\text{CTL}$ and to each other.

4 Model checking

4.1 Model checking $\text{CCTL}_{C_0}$ and $\text{CCTL}_{C_1}$

It turns out that model-checking $\text{CCTL}_{C_0}$ is polynomially equivalent to model-checking $\text{CCTL}_{C_1}$ (or $\text{CTL}$), as both problems are $\text{P}$-complete.

Theorem 9. The model-checking problem for $\text{CCTL}_{C_1}$ is $\text{P}$-complete.

Proof. $\text{P}$-hardness comes from the $\text{P}$-hardness of $\text{CTL}$ model-checking. For membership in $\text{P}$, we provide polynomial-time procedures to deal with the subformulas $\text{E} \psi U[C] \psi'$ and $\text{A} \psi U[C] \psi'$ with $C = \Sigma_{i=1}^k \varphi_i \sim k$. Consider a Kripke structure $S = (Q, R, \ell)$, and inductively assume that the truth values of $\psi, \psi'$
and $\varphi_i$ over each state of $S$ are known: these sub-formulas will be seen as atomic propositions in the following.

To each state $q$ occurring along a path, we associate a cost $|q|_C = |\{ i \mid q \models \varphi_i \}|$, and note that the value of $|q|_C$ is in $O(|C|)$. This cost is additively extended to paths in the usual way. Deciding the truth value of the path formula $\psi U_{|C|} \psi'$ over any path $\rho$ verifying $\psi U \psi'$ then amounts to checking whether there exists a finite prefix $\rho' \cdot q$ of $\rho$ such that $|\rho'|_C \sim k$, $q \models \psi'$ and $\forall i \leq |\rho'|, \rho'(i) \models \varphi$.

We reduce this problem to the model-checking problem for $\Delta^2_P$ (DKS with 0/1-durations) for which there exists a polynomial-time algorithm [15]. We build from $S$ a DKS $\langle S', R', \ell' \rangle$ as follows: for each state $q \in Q$ with $|q|_C = n$, $Q'$ contains $n + 1$ additional states $q_0, \ldots, q_n$. $R'$ is then defined as \{ $q \xrightarrow{0} q_0 \mid q \in Q \} \cup \{ q_i \xrightarrow{1} q_{i+1} \mid q \in Q, i < |q|_C \} \cup \{ q_0 \xrightarrow{0} q' \mid (q, q') \in R, n = |q|_C \}$. Finally, we set $\ell'(q_i) = \emptyset$ for all $q_i \in Q' \setminus Q$ and $\ell'(q) = \ell(q) \cup \{ \text{ok} \}$ for all $q \in Q' \cap Q$, where $\text{ok}$ is a new atomic predicate.

To each path $\rho = q\sigma \sigma' q'$ in $S$, we associate the path $\bar{\rho} = q q_0 \ldots q_0 \bar{\sigma} q'$ in $S'$. It can now be shown by induction on run lengths that $\rho$ satisfies $\psi U_{|C|} \psi'$ if and only if $\bar{\rho}$ satisfies the $\text{TCTL}$ path formula $(\text{ok} \Rightarrow \psi) U_{|\bar{k}|} (\text{ok} \land \psi')$. □

Since $\text{CCTL}_{C_0}$ includes CTL and is included in $\text{CCTL}_{C_1}$, we get:

**Corollary 1.** The model-checking problem for $\text{CCTL}_{C_0}$ is $\text{P}$-complete.

### 4.2 Model-checking $\text{CCTL}_{B(C_0)}$ and $\text{CCTL}_{aC_1}$

We now establish the $\Delta^2_P$-completeness of model-checking for the fragments $\text{CCTL}_{B(C_0)}$, $\text{CCTL}_{aC_1}$, and $\text{CCTL}_{B(aC_1)}$. Let us first recall the definition of the complexity class $\Delta^2_P$, one of the classes of the polynomial hierarchy.

**Definition 10.** $\Delta^2_P = \text{P}^{\text{NP}}$ is the class of problems solvable in polynomial time with access to an oracle for some $\text{NP}$-complete problem.

We now prove $\Delta^2_P$-hardness of the model-checking problem for $\text{CCTL}_{B(C_0)}$.

**Theorem 11.** The model-checking problem for $\text{CCTL}_{B(C_0)}$ is $\Delta^2_P$-hard.

**Proof.** We proceed by reduction from the $\Delta^2_P$-complete problem $\text{SNSAT}$ [12].

Given $p$ families of variables $X_1, \ldots, X_p$ with $X_i = \{ x_i^1, \ldots, x_i^{m_i} \}$ and a set $Z = \{ z_1, \ldots, z_p \}$ of $p$ variables, an instance $I$ of $\text{SNSAT}$ is defined as a collection of $p$ propositional formulas $\varphi_1, \ldots, \varphi_p$ under 3-conjunctive normal form (3-CNF), where each $\varphi_i$ involves variables in $X_i \cup \{ z_1, \ldots, z_{i-1} \}$, and the values of each $z_i$ is defined as $z_i \equiv 3 \in I$, $\varphi_i(z_1, \ldots, z_{i-1}, X_i)$. The instance $I$ is positive if the value of $z_p$ is $\top$. We denote by $v_I$ the unique valuation of variables in $Z$ induced by $I$.

From $I$, we define the KS described in Figure 1. Every state is labeled by its name, and in addition every state $\bar{z}_i$ is labeled by some new atomic proposition $P_z$. We use $X$ to denote the set $X_1 \cup \cdots \cup X_p$ and $\bar{X}$ for $X \cup Z$. A path $\rho$ from $q_p$ to $q_F$ describes the valuation $v_\rho$, such that $v_\rho(y) = \top$ if $\rho$ visits state $y$ and $\bot$ if it visits $\bar{y}$ for every variable $y$ in $\bar{X}$. We use a $\text{CCTL}_{B(C_0)}$ formula to ensure
Theorem 14. The model-checking problem for where \( \tilde{\varphi} \) be a DKS. Let \( W \) be the set of weights occurring in specifications over Durational Kripke structures. CCTL formulas allow to deal with the cost (or duration) of paths (i.e. the sum of the weight of every transition occurring along the path). This problem is \( \Delta_2^p \)-complete [13]. Let \( S = (Q, R_S, \ell) \) be a DKS. Let \( W \) be the set of weights occurring in \( S \). We define the Kripke structure \( S' = (Q', R_S, \ell') \) as follows:

- \( Q' \models \equiv Q \cup \{ (q, d, q') \mid \exists(q, d, q') \in R_S \}, \)
- for any \( (q, d, q') \in R_S \), we add \( (q, (q, d, q')) \) and \( ((q, d, q'), q') \in R_S \); and
- \( \ell' : Q' \rightarrow \mathbb{2}^{AP'} \) with \( AP' \equiv AP \cup \{ ok \} \cup \{ P_d \mid d \in W \} \) — we assume \( ok, P_d \not\in AP \).

And we have: \( \ell'(q) \equiv \ell(q) \cup \{ ok \} \) for any \( q \in Q \), and \( \ell'(q, d, q') = \{ P_d \} \).

Now we can easily see that \( q \models_S \Phi \) with \( \Phi \in \mathcal{C}TL \) is equivalent to \( q \models_{S'} \tilde{\Phi} \), where \( \tilde{P} \equiv P, \tilde{\not\psi} \equiv \not\tilde{\psi}, \tilde{\varphi} \land \tilde{\psi} \equiv \tilde{\varphi} \land \tilde{\psi}, \tilde{E}_{\varphi}U_{\sim \psi} \equiv \tilde{E}_{\varphi}U_{\sim \psi}, \tilde{E}_{\varphi}U_{\sim \psi} \equiv \tilde{E}_{\varphi}U_{\sim \psi} \) and \( \tilde{A}_{\varphi}U_{\sim \psi} \equiv A(ok \Rightarrow \tilde{\varphi})U_{\sim \psi} \) with \( C(\sim c) \equiv \sum_{d \in W} \ell'(q, d, q') \).

Theorem 14. The model-checking problem for \( \mathcal{C}CTL_{R(S)} \) is in \( \Delta_2^p \).
Proof (sketch). Let $\mathcal{S} = \langle Q, R, \ell \rangle$ be a KS. For this proof, we only need to provide NP procedures to deal with sub-formulas of the form $\EF[\alpha][C]\psi$ and $\EG[\alpha][C]\psi$. First let $\{C_1, \ldots, C_l\}$ be the set of $\alpha\mathcal{C}_l$ constraints occurring in $C$. Each $C_i$ is of the form $\sum_{j \in [l]} \alpha_j^i \cdot \varphi_j^i \sim d_i$. And let $d_{\max}$ be the maximal integer constant occurring in $C$. Now we can present the algorithms:

- $\Phi \stackrel{\text{def}}{=} \EF[\alpha][C]\psi$: If $q \models \Phi$, then there exists a run $\rho$ starting from $q$ and leading to some $q'$ such that (1) $q' \models \psi$ and (2) $\rho$ without $q'$ satisfies the constraint $C$. First note that we can assume that the length of $\rho$ is bounded with respect to the model and formula: a sequence of $|Q|$ states contributes for at least 1 to some linear expressions in $C$ and then the length of $\rho$ is in $O(|Q|2^{|C|})$ due to the binary encoding of the constants. An easy NP algorithm consists in guessing the Parikh image of the transitions in $\rho$, which can be represented in polynomial size. Moreover it is possible to check (in polynomial time) that $q'$ satisfies $\psi$, $\rho$ without $q'$ satisfies $C$, and $F_\rho$ corresponds to a path in $\mathcal{S}$.

- $\Phi \stackrel{\text{def}}{=} \EG[\alpha][C]\psi$: For this case we have to find an infinite path $\rho$ satisfying the property “if the current prefix satisfies the constraint $C$, then the next state has to satisfy $\psi$”. Every constraint $C_i \in \alpha\mathcal{C}_l$ in $C$ may change its truth value at most twice along $\rho$. Therefore $\rho$ can be decomposed in a bounded number of parts over which the truth value of every constraint is constant. As previously, the length of every part is bounded and its Parikh image can be encoded in polynomial size. Moreover it is possible to ensure that the juxtaposition of all $\rho_m$ is correct.

A direct corollary of Theorems 11, 13 and 14 is:

**Corollary 2.** The model-checking problem for $\text{CCTL}_C$ is $\Delta^p_p$-complete for each $C \in \{\alpha\mathcal{C}_l, B(\mathcal{C}_0), B(\mathcal{C}_1), B(\mathcal{C}l)\}$.

### 4.3 Diagonal constraints

We now show that even if diagonal constraints lead to strictly more expressive logics than CTL, model checking $\text{CCTL}_C$ and $\text{CCTL}_{\mathcal{C}_l}$ is not more difficult than model checking CTL itself.

**Theorem 15.** The model-checking problem for $\text{CCTL}_{\mathcal{C}_l}$ is $\mathcal{P}$-complete.

Proof (sketch). $\mathcal{P}$-hardness comes from that of $\text{CCTL}_{\mathcal{C}_0}$ model-checking. Using the fact that $A\varphi'[U[\alpha][C]]\psi' \equiv A\EF[\alpha][C\land\varphi'=0]\psi' \equiv \neg\EG[\alpha][C\land\varphi'=0]\neg\psi'$, to show membership in $\mathcal{P}$, we only need to provide polynomial-time procedures to verify sub-formulas of the form $\EF[\alpha][C]\psi'$ and $\EG[\alpha][C\land\varphi'=0]\psi'$ with $C \equiv \varphi - \psi \sim k$. Consider a Kripke structure $\mathcal{S} \stackrel{\text{def}}{=} \langle Q, R, \ell \rangle$. As previously, we associate a “cost” to each state $q \in Q$. In this case however, $|q|_{\mathcal{C}_l}$ can only be -1, 0 or 1 depending on the truth values of $\varphi$ and $\psi$. Inductively assume that the truth values of $\varphi$, $\psi$, $\varphi'$ and $\psi'$ over each state of $\mathcal{S}$ are known: these sub-formulas will be seen as atomic propositions in the following. We distinguish the two main cases below:
- \( \Phi \defeq E_{\varphi'} \cup [C] \psi' \): We consider the weighted and directed graph \( G_S = (V, E) \) representing the transition relation of \( S \) restricted to states verifying the formula \( E_{\varphi'} \cup \psi' \), where edges are weighted by the cost of their source state and where only edges whose source verifies \( \varphi' \) are considered.

If \( C \defeq \#\varphi - \#\psi \leq k \), then the formula holds true on state \( q \) if and only if there exists a state \( q' \) such that \( q' \models S \psi' \) and either an elementary (i.e. acyclic) path \( \rho \) in \( G_S \) of weight less than \( k \) from \( q \) to \( q' \), or a path from \( q \) to some state \( q'' \) appearing on a negative-weight cycle, and from \( q'' \) to \( q' \). Using a classical reachability algorithm over \( G_S \), one can determine the existence of such paths in polynomial time.

If \( C \defeq \#\varphi - \#\psi = k \) with \( k \geq 0 \), we will compute the relation \( R_k \) over \( V^2 \) denoting the existence of a run of weight \( k \) between states \( q \) and \( q' \) and simply test whether \( (q, q') \in R_k \) for some \( q' \) verifying \( \psi' \). Using dichotomy and simple fixed-point computations, we are able to compute \( R_k \) in time polynomial in \( |\Phi| \), i.e. logarithmic in \( k \).

The treatment of negative weights is omitted.

Corollary 3. The model-checking problem for \( \text{CCTL}_{C_3} \) is \( \mathcal{P} \)-complete.

Proof (sketch). In this setting, each state contributes to the cost of a path by a certain positive or negative number whose absolute value is bounded by some integer \( d \). Similarly to the technique used in the proof of Theorem 9, the idea is to build a durational Kripke structure, this time with weights in \( \{−1, 0, 1\} \), by adding intermediate states. Once this DKS is built, relations \( R_i \), \( R_i^+ \) and \( R_i^- \) may be computed as previously, and the satisfaction of the formula under consideration tested.

Theorem 16. The model-checking problem for \( \text{CCTL}_{B(C_2)} \) is undecidable.

Proof (sketch). This is done by reduction from the halting problem of a two-counter machine \( M \) with counters \( C \) and \( D \). We define a Kripke structure \( S_M \) with one state to simulate each of \( M \)'s instructions, plus some auxiliary states. We use labels \( \varphi_X^+ \) and \( \varphi_X^- \) with \( X \in \{C, D\} \) to witness increments and decrements, and additional labels \( \text{ok}_X \), \( \text{ko}_X \) to simulate the positive test \( "X = 0" \): whenever the counter’s value is assumed to be zero, and before simulating the next instruction, the run goes through an auxiliary state labeled \( \text{ko}_X \) whose unique successor is labeled \( \text{ok}_X \). Hence along any run in \( S_M \), a prefix satisfies \( \#\text{ko}_X > \#\text{ok}_X \) right after counter \( X \) was deemed equal to zero, and only then. By counting occurrences of these predicates, one can write a \( \text{CCTL}_{B(C_2)} \) formula expressing the fact that \( M \) is correctly simulated by \( S_M \) and never halts.
5 Freeze variables

Instead of using counting constraints associated with temporal modalities, we now consider freeze variables and explicit constraints inside formulas.

Definition 17. Given a set of atomic propositions \( AP \) and a set of variables \( V \), we define: \( CCTL^\psi \supseteq \phi, \psi ::= P \mid \phi \land \psi \mid \neg \phi \mid z[\psi].\phi \mid C \mid E\phi U\psi \mid A\phi U\psi \) where \( P \in AP \) and \( C \) is a constraint \( \sum_{i=1}^{l} \alpha_i \cdot z_i \sim c \) with \( z_i \in V \), \( \alpha_i \in \mathbb{N} \), \( c \in \mathbb{N} \) and \( \sim \in \{<,\leq,=,\geq,>\} \).

Intuitively \( z[\psi].\phi \) means that (1) the variable \( z \) is reset to zero and associated with the sub-formula \( \psi \) (i.e. \( z \) will evolve like \( \psi \) in the future) and (2) given this semantics for \( z, \phi \) holds for the current state. We say that an occurrence of some variable \( z \) is free in \( \phi \) when this occurrence does not appear in the scope of a reset operator \( \to \); a formula without any free variable is closed. For example, the \( CCTL_{(C)} \) formula \( EF[5P \leq \pi A\pi' > 2]P' \) can be expressed in \( CCTL^\psi \) as the formula \( z[P].z'[P'].EF(5 \leq z' > 2 \land P') \).

A \( CCTL^\psi \) formula \( \phi \) is interpreted in a state of a KS extended with a valuation for any free variable in \( \phi \) and an environment associating a sub-formula to any free variable. We use \( dom \) to denote the domain of such functions and \( \bot \) to represent undefined values. Given a function \( f \) and \( x \in dom(f) \), we use \( f[x \leftarrow a] \) to denote the function mapping \( x \) to \( a \), and every element \( y \) to \( f(y) \) if \( y \neq x \). Finally let \( SubF(\phi) \) be the set of all \( \phi \) sub-formulas.

Given a valuation \( v : V \rightarrow \mathbb{N} \cup \{\bot\} \) for a set of variables occurring in a \( CCTL^\psi \) formula \( \phi \), and an environment \( \varepsilon : V \rightarrow SubF(\phi) \cup \{\bot\} \) such that \( dom(v) = dom(\varepsilon) \), and given a finite run \( \pi \), we define the valuation \( (v + _\pi \varepsilon) \) as follows: \( (v + _\pi \varepsilon)(z) \overset{\text{def}}{=} \bot \) if \( z \notin dom(v) \), and otherwise \( (v + _\pi \varepsilon)(z) \overset{\text{def}}{=} v(z) + [j|0 \leq j \leq |\pi| \land \pi(j) = \varepsilon(z)] \). The semantics of \( CCTL^\psi \) is defined as follows:

Definition 18. The following clauses define when a state \( q \) of some KS \( S = (Q,R,\ell) \) and a valuation \( v \) satisfy a \( CCTL^\psi \) formula \( \phi \) in an environment \( \varepsilon \) - written \( (q,v) \models_{S,\varepsilon} \phi \) - by induction over the structure of \( \phi \) (we omit the cases of Boolean modalities):

\[
(q,v) \models_{S,\varepsilon} E\phi U\psi \iff \exists \rho \in \text{Runs}(q), v \models_{\rho,\varepsilon} \phi U\psi \\
(q,v) \models_{S,\varepsilon} A\phi U\psi \iff \forall \rho \in \text{Runs}(q), v \models_{\rho,\varepsilon} \phi U\psi \\
(q,v) \models_{S,\varepsilon} z[\psi].\phi \iff (q,v[z \leftarrow 0]) \models_{S,\varepsilon[z \leftarrow \psi]} \phi \\
(q,v) \models_{S,\varepsilon} \sum_{i=1}^{l} \alpha_i \cdot z_i \sim c \iff \sum_{i=1}^{l} \alpha_i \cdot v(z_i) \sim c
\]

where \( v \models_{\rho,\varepsilon} \phi U\psi \iff \exists i \geq 0, (\rho(i),v + _\varepsilon \rho(i-1)) \models_{S,\varepsilon} \phi \) and \( \forall 0 \leq j < i, (\rho(j),v + _\varepsilon \rho(j-1)) \models_{S,\varepsilon} \phi \).

Theorem 19. Model checking closed \( CCTL^\psi \) formulas is \( \text{PSPACE}\)-complete.

Proof. \( \text{PSPACE}\)-hardness can be proved by a reduction from \( \text{QBF}\). \( \text{PSPACE}\)-membership is obtained by considering a non-deterministic algorithm working
in polynomial space to decide whether a $\text{CCTL}^v$ formula holds for a state $q$ within a KS $S$. The main idea is to encode a configuration $(q, v, \varepsilon)$ in polynomial size: this is possible for $v$ since we just have to record the value for the counter $z$ up to $d_{\text{max}} + 1$ where $d_{\text{max}}$ is the maximal constant used in a constraint with $z$. In order to verify $E\varphi U\psi$ — we assume that $\varphi$ and $\psi$ have already been treated — we guess, from the current configuration $(q, v, \varepsilon)$, the next configuration $(q', v', \varepsilon)$ and then we verify that there is a transition in $S$ leading from $q$ to $q'$ such that the valuation $v$ is updated with $v'$ w.r.t. the environment $\varepsilon$. Then it remains to verify that either $\psi$ or $\varphi$ holds for $(q', v')$ (and in the latter case, guess a new configuration etc.). The same holds for $\text{EG}$. The operator $z[\psi].\varphi$ changes the environment $\varepsilon$ and resets $z$ to zero. And for any configuration one can decide the truth value of a constraint $C$.

\[\square\]

6 Conclusion

In several cases (up to $B(\alpha C_1)$ constraints) the logics we introduce are not more expressive than $\text{CTL}$, but can concisely express properties which would be difficult to write in that logic. In particular, $\text{CCTL}_{C_0}$ and $\text{CCTL}_{B(C_0)}$ can be exponentially more succinct than $\text{CTL}$. As for the remaining fragments, even though $\text{CCTL}_{C_2}$ is strictly more expressive than $\text{CTL}$, model-checking remains polynomial up to $\text{CCTL}_{C_3}$ (complexity results are summarized in Figure 2). Further work on $\text{CCTL}$ will include completing the study of succinctness of its fragments with respect to each other and to other logics, looking for an upper complexity bound for the model-checking of $\text{CCTL}_{\alpha C_1}$, as well as investigating new kinds of constraints and extensions to $\text{LTL}$ and $\text{CTL}^*$. 

![Fig. 2. Summary of model-checking complexity results.](image-url)
References