

The Satisfiability Problem of Temporal-Spatial Logics over Quasi-Temporal Graphs

Eric Alsmann 

Theoretical Computer Science / Formal Methods, University of Kassel, Germany

Martin Lange 

Theoretical Computer Science / Formal Methods, University of Kassel, Germany

Igor Semezie 

Theoretical Computer Science / Formal Methods, University of Kassel, Germany
Laboratoire Méthodes Formelles, ENS Paris-Saclay, France

Abstract

Temporal Graph Neural Networks (TGNN) are used to detect patterns in so-called temporal graphs (TG): graphs in which edges may be added or removed over time. Sälzer et al. suggested to study the expressive power of TGNNs through temporal-spatial logics, specifically the combination of the linear-time temporal logic LTL and modal logic K. In this paper we investigate the computational complexity of the satisfiability problem for this logic and its natural extension in which the LTL part is replaced by a linear-time μ -calculus, reflecting TGNNs' ability to recognise not just star-free (word) languages. We consider their interpretation over a more natural class of models which we call quasi-temporal graphs (QTG), relaxing certain conditions on the temporal evolutions of nodes that are indifferent to the logic. We formalise this by giving an adjusted notion of bisimulation which preserves satisfaction in these logics. It turns out that satisfiability over the class of QTGs is PSPACE-complete for the temporal-spatial logic based on LTL but becomes EXPTIME-complete when based on the μ -calculus. This is in contrast to the situation on words where both logics are PSPACE-complete. We also discuss consequences for the special satisfiability problems over the class of TGs.

2012 ACM Subject Classification Theory of computation $\rightarrow$ Modal and temporal logics; Theory of computation $\rightarrow$ Machine learning theory

Keywords and phrases linear-time temporal logic, modal logic, temporal graphs, computational complexity, automated reasoning

Digital Object Identifier 10.4230/OASICS.TIME.2026.8

1 Introduction

Machine learning models are computational devices that have enabled revolutionary developments and success in artificial intelligence tasks with numerous applications in all kinds of industrial, societal, scientific etc. domains. One such model which has gained recent interest because of its use in the analysis of social networks for example, is that of a *Temporal Graph Neural Network* (TGNN) [25, 33]. It builds upon the better known model of *Graph Neural Network* (GNN) [18] which can be used to recognise patterns in static graphs, to the recognition of properties of graphs that are dynamically evolving in time.

Sälzer et al. [30] suggested to use product logics as yardsticks for the expressiveness of TGNNs, specifically two-dimensional products where the operators of one dimension are used to navigate through the sequence – called the *temporal* dimension – and the operators of the other – called the *spatial* dimension – are used to navigate through a single graph. This corresponds closely to the functionality of TGNNs. Formally, a TGNN recognises *temporal graphs* (TG) which are finite sequences of graphs which share the same node set. A TGNN processes such a sequence by iteratively updating node representations, combining



© Eric Alsmann, Martin Lange, and Igor Semezie;
licensed under Creative Commons License CC-BY 4.0

33rd International Symposium on Temporal Representation and Reasoning (TIME 2026).

Editors: AndreA Orlandini and Sophie Pinchinat; Article No. 8; pp. 8:1–8:16

OpenAccess Series in Informatics



OASICS Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

spatial information propagated through graph edges within each graph and with temporal information carried forward from preceding time steps. Crucially, spatial information is aggregated only from nodes within a bounded neighbourhood, while temporal information is passed on recursively in most TGNNs, so that the representation of a node at step $i + 1$ depends on its own representation at step i .

A natural candidate for a two-dimensional logic to capture the expressiveness of TGNNs is $\text{LTL}_f \times K$, using Linear-Time Temporal Logic on finite traces (LTL_f) [17] in the temporal dimension and the standard modal logic K in the spatial dimension. This provides some limited recursive mechanism temporally and no recursive mechanism spatially, i.e. expressible properties can address moments arbitrarily far in the future, but within a graph they can only speak about a neighbourhood of a node of fixed radius. This corresponds exactly to the way that TGNN collect information about a temporal graph, at least when abstracting away from the difference between real-valued and Boolean-valued information retrieval.

A natural extension of $\text{LTL}_f \times K$ is obtained when LTL is replaced by the linear-time μ -calculus μTL [2, 35], known to exceed the expressiveness of LTL by capturing all regular properties [11]. This is also motivated by the question after the formal expressiveness of TGNNs, as the use of non-linear functions in them permits them to do modulo counting, i.e. to recognise temporal graphs of even length for example. Hence, $\text{LTL}_f \times K$ may be a natural logic to consider with respect to TGNN expressiveness from a pragmatic point of view due to the popularity and simplicity of LTL ; but a logic based on μTL instead seems to be better suited from a technical perspective in order to fully capture the expressiveness of TGNNs.

As with LTL , μTL can of course be interpreted purely on finite words only, leading to the logic μTL_f [24] which comes with less combinatorial complexity. In particular, there is no need to distinguish least and greatest fixpoint operators when formulas are guarded, using the same kind of reasoning as is possible for the modal μ -calculus on acyclic structures [28]. Asymptotically, all four logics – LTL and μTL , interpreted on finite or infinite words – share the same complexity: their satisfiability problems are PSPACE-complete [32, 35].

Structure and contribution. We study the computational complexity of logics suggested for temporal graphs, in particular the aforementioned $\text{LTL}_f \times K$ and $\mu\text{TL}_f \times K$. In Sect. 2, we formally introduce (quasi-)temporal graphs and the logics $\text{LTL}_f \times K$ and $\mu\text{TL}_f \times K$. In Sect. 3 we show that their properties are invariant under bisimulations, adapted suitably to the temporal-spatial setting, while the class of TGs is not, thus justifying the use of the relaxed model of *quasi*-temporal graphs (QTG). In Sect. 4 we study the complexity of satisfiability for $\text{LTL}_f \times K$ and $\mu\text{TL}_f \times K$ over QTGs, obtaining the perhaps surprising result that – whilst all the ingredients LTL_f , μTL_f and K are PSPACE-complete, $\text{LTL}_f \times K$ remains PSPACE-complete but $\mu\text{TL}_f \times K$ becomes EXPTIME-complete. In Sect. 5 we conclude with remarks on the complexity of satisfiability over the class of TGs and the expressiveness of TGNNs, as well as further work for the future.

Related work. Logical characterisations of static GNNs via modal logics [3, 29] have inspired analogous questions for temporal architectures. For TGNNs, no connection to logics had been established prior to [30], where three classes of TGNNs are characterised via two-dimensional product logics of temporal and modal logics. Questions regarding the exact complexity of their satisfiability problems as well as their ability to distinguished temporal graphs from non-temporal graphs were not investigated there.

Many-dimensional logics [16, 27] offer a general framework for combining modal logics, each capturing a distinct aspect of reasoning such as time, space, or knowledge. The standard theory distinguishes *fusion* logics, in which the component logics interact only through

shared propositional variables, from *product* logics, in which models carry a Cartesian product structure. Satisfiability and axiomatisation of two-dimensional product logics have been studied in depth by Gabbay et al. [16]. There, product logics are, however, interpreted on structures with *static* edge sets, in fact *products* of structures, which enforces the commutativity axioms $\Diamond \bigcirc p \leftrightarrow \bigcirc \Diamond p$ and $\bigcirc \Box p \rightarrow \Box \bigcirc p$. These are not axioms anymore when edges may change over time.

The work that is most closely related to the present study is on temporal description logics, cf. [26]. The structures known as temporal graphs in the machine learning community are known as “temporal interpretations over a constant domain” in the description logic setting, although they are fairly simple special cases with a single role (binary relation) only. The logic $\text{LTL} \times \text{K}$, as considered here, could be seen as a simple description logic over such structures. A minor difference lies in the treatment of time: while the time line there is considered to be infinite, it is finite in the study here, hence we consider $\text{LTL}_f \times \text{K}$. The main difference, though, is that we also study an extension in the temporal dimension, namely the logic $\mu\text{TL}_f \times \text{K}$, which is not done in the description logic work, and there the focus lies on different reasoning problems where complexity is introduced via TBox axioms which is not present here.

2 Preliminaries

Let AP be a countable set of atomic propositions and $\mathcal{V}$ be a countable set of (monadic second-order) variables. A *graph* is a triple $(S, \rightarrow, L)$, where S is a finite set, $\rightarrow \subseteq S \times S$ is the *transition relation*, and $L : S \rightarrow 2^{\text{AP}}$ is the *labeling of the nodes* such that $|L(v)| < \infty$ for all $v \in S$.

(Quasi-)Temporal graphs. A *quasi-temporal graph* (QTG) is a tuple $\mathfrak{G} = (G_0, \sigma_0, G_1, \sigma_1, \dots, \sigma_{n-2}, G_{n-1})$, such that for all $0 \leq i < n$, $G_i = (S_i, \rightarrow_i, L_i)$ is a directed, 2^{AP} -labeled graph, and for all $0 \leq i < n-1$, $\sigma_i : S_i \rightarrow S_{i+1}$ is the *i-th time successor function* which is partial in general. A *pointed QTG* is a triple $(\mathfrak{G}, v, i)$ where $\mathfrak{G}$ is a QTG as above, $i \in \{0, \dots, n-1\}$ and $v \in S_i$. Thus, it simply singles out a particular node at a particular moment in time in the QTG. Intuitively, $\sigma_i(v)$ is what v becomes in the *i-th* time step. Nodes need not have temporal successors, but if they do, then these are unique. This is why each σ_i is a partial function. A node at moment $i > 0$ can be the temporal successor of more than one node at moment $i-1$, though. We introduce some useful notation. Let $\mathfrak{G}$ be as above. For i, j with $0 \leq i < j < n-1$ we write $\sigma_{i,j}$ for the composition $\sigma_i \circ \dots \circ \sigma_{j-1}$. Here, composition of partial functions can be seen as relation composition. In particular, nodes in G_i need not have a $\sigma_{i,j}$ -successor for $j > i$ but if they do then it is the unique node that is obtained by following the functions $\sigma_i, \dots, \sigma_{j-1}$ successively starting from $v \in S_i$. Moreover, in this case we have $\sigma_{i,j}(v) \in S_j$ unless it does not exist.

A *temporal graph* (TG) is a quasi-temporal graph $\mathfrak{G} = (G_0, \sigma_0, G_1, \sigma_1, \dots, \sigma_{n-2}, G_{n-1})$ in which each σ_i is total and bijective for $i = 0, \dots, n-2$. This is clearly equivalent to the formulation found in the literature which defines a temporal graph as a sequence of edge relations over the same nodes set.

Linear-time temporal logic for temporal graphs. Let AP be as above. Formulas of $\text{LTL}_f \times \text{K}$ over AP are defined by the grammar

$$\varphi ::= p \mid \varphi \wedge \varphi \mid \neg \varphi \mid \bigcirc \varphi \mid \varphi \mid \varphi \cup \varphi \mid \Diamond \varphi$$

where $p \in \text{AP}$. *Next* ($\bigcirc$) and *until* (U) are the *temporal* operators, and *Diamond* ($\Diamond$) is the spatial operator. The set $\text{sub}(\varphi)$ of subformulas of an $\text{LTL}_f \times \text{K}$ formula φ is defined straightforwardly. The *size* of φ is $|\varphi| := |\text{sub}(\varphi)|$.

We define the usual Boolean, temporal and modal abbreviations as follows.

$$\begin{array}{lll} \text{ff} := p \wedge \neg p \quad \text{for some } p \in \text{AP} & \text{tt} := \neg \text{ff} & \Box \varphi := \neg \Diamond \neg \varphi \\ \varphi \vee \psi := \neg(\neg \varphi \wedge \neg \psi) & & \odot \varphi := \neg \bigcirc \neg \varphi \\ \varphi \text{ R } \psi := \neg(\neg \varphi \text{ U } \neg \psi) & \text{F } \varphi := \text{tt U } \varphi & \text{G } \varphi := \neg \text{F } \neg \varphi \end{array}$$

An $\text{LTL}_f \times \text{K}$ formula is in *negation normal form* (NNF) if it is built from literals p and $\neg p$ using the operators $\wedge, \vee, \Diamond, \Box, \bigcirc, \odot, \text{U}$ and R only. It is standard to show that every formula can equivalently be transformed into NNF with an at most linear blow-up in size. Equivalence needs a formal notion of semantics of course, which we define next.

Formulas of $\text{LTL}_f \times \text{K}$ are interpreted on pointed QTG $(\mathfrak{G}, v, i)$ with $\mathfrak{G} = (G_0, \sigma_0, \dots, \sigma_{n-2}, G_{n-1})$ and $G_i = (S_i, \rightarrow_i, L_i)$ for $i = 0, \dots, n-1$ as follows.

$$\begin{array}{ll} \mathfrak{G}, v, i \models p & \iff p \in L_i(v) \\ \mathfrak{G}, v, i \models \varphi \wedge \psi & \iff \mathfrak{G}, v, i \models \varphi \text{ and } \mathfrak{G}, v, i \models \psi \\ \mathfrak{G}, v, i \models \neg \varphi & \iff \mathfrak{G}, v, i \not\models \varphi \\ \mathfrak{G}, v, i \models \bigcirc \varphi & \iff i < n-1 \text{ and } \mathfrak{G}, \sigma_i(v), i+1 \models \varphi \\ \mathfrak{G}, v, i \models \varphi \text{ U } \psi & \iff \text{there is } k \text{ s.t. } i \leq k < n \text{ and } \mathfrak{G}, \sigma_{i,k}(v), k \models \psi \\ & \quad \text{and for all } j \text{ with } i \leq j < k : \mathfrak{G}, \sigma_{i,j}(v), j \models \varphi \\ \mathfrak{G}, v, i \models \Diamond \varphi & \iff \text{there is } u \in S_i \text{ s.t. } v \rightarrow_i u \text{ and } \mathfrak{G}, u, i \models \varphi \end{array}$$

Two formulas φ, ψ of $\text{LTL}_f \times \text{K}$ are equivalent, written $\varphi \equiv \psi$, if for all pointed QTGs $\mathfrak{G}, v, i$ we have $\mathfrak{G}, v, i \models \varphi$ iff $\mathfrak{G}, v, i \models \psi$.

The following is standard knowledge about LTL, resp. LTL_f , and holds in $\text{LTL}_f \times \text{K}$ for exactly the same reasons.

► **Lemma 1.** *For all $\varphi, \psi \in \text{LTL}_f \times \text{K}$ we have*

$$\varphi \text{ U } \psi \equiv \psi \vee (\varphi \wedge \bigcirc(\varphi \text{ U } \psi)) \quad , \quad \varphi \text{ R } \psi \equiv \psi \wedge (\varphi \vee \odot(\varphi \text{ R } \psi)) \quad .$$

Fixpoint logic for temporal graphs. The two-dimensional logic $\mu\text{TL}_f \times \text{K}$ is obtained from $\text{LTL}_f \times \text{K}$ by replacing the temporal Until operator with a general fixpoint operator rec . Traditionally, μ -calculi are introduced with two different fixpoint operators μ and ν to denote the generally different *least* and *greatest* fixpoint of a predicate transformer $\varphi(X)$ making use of the Knaster-Tarski Theorem [21, 34]. However, the difference only shows up in limits, i.e. in infinite unfoldings. This is prevented here by a combination of the finiteness in the temporal dimension of models and the impossibility to explore paths of infinite length inside a single graph, ensured by the syntactic restriction of guardedness.

Let AP be given as above. Moreover, let $\text{Var} = \{X, Y, \dots\}$ be a countable set of monadic second-order variables. Formulas of $\mu\text{TL}_f \times \text{K}$ over AP (and, technically, also over Var) are defined by the grammar

$$\varphi ::= p \mid X \mid \varphi \wedge \varphi \mid \neg \varphi \mid \bigcirc \varphi \mid \text{rec } X. \varphi \mid \Diamond \varphi$$

where $p \in \text{AP}$ and $X \in \text{Var}$.

The set $\text{sub}(\varphi)$ of subformulas of φ is defined as usual. Again, the size of φ is $|\varphi| := |\text{sub}(\varphi)|$. As with $\text{LTL}_f \times K$, we introduce $\mathbf{ff}, \mathbf{tt}, \vee, \odot$ and $\square$ as abbreviations. A $\mu\text{TL}_f \times K$ formula is said to be in NNF, if it is built from literals p and $\neg p$ for any $p \in \text{AP}$, as well as variables $X \in \text{Var}$, using only the operators $\wedge, \vee, \odot, \odot, \text{rec}, \diamond$ and $\square$. Again, it is standard to show that every formula can equivalently be transformed into one in NNF with an at most linear blow-up, with equivalence formally defined below.

The operator rec acts as a *binder* for the variable X . We only consider sentences, i.e. formulas without unbound occurrences of variables, as top-level formulas. Formulas with free variables can of course occur as subformulas of a formula in $\mu\text{TL}_f \times K$.

We employ further syntactic restrictions on formulas. They are all standard in the literature on μ -calculi, cf. [6, 7, 12]. (I) Formulas must be *well-named*: each variable X gets bound at most once in any subformula of the form $\text{rec } X.\psi$. Then such a φ comes with an identifier function fp_φ mapping a variable X that is bound in φ to the unique subformula $\text{rec } X.\psi$ in which it is bound. (II) There must be an even number of negation symbols on any path in the syntax tree of a formula φ , starting in $fp_\varphi(X)$ for any $X \in \text{Var}$ and ending in a leaf node X . This ensures that fixpoints are well-defined because the underlying predicate transformers $\psi(X)$ are indeed monotone. (III) Formulas must be *temporally guarded*. In the syntax tree there must be at least one occurrence of an operator $\odot$, on any path starting in $fp_\varphi(X)$ and ending in a leaf node X , for any variable X .

Unlike (I) and (II), restriction (III) is in fact a semantic restriction. One cannot simply use the well-known fact that formulas of the μ -calculus can equivalently be transformed into guarded form [22, 37, 8], because here we require variables in fixpoint formulas not just to be guarded by *some* modal or temporal operator, but specifically by a temporal one. Hence, formulas like $\text{rec } X.p \vee \diamond X$ are guarded from a modal μ -calculus perspective but temporally unguarded regarding condition (III). Such a formula expresses reachability of a p -node in the current graph at an arbitrary distance, and this is a property that TGNN most likely cannot check which justifies the restriction to temporally guarded formulas. We leave it as an open question whether equivalence of an unrestricted formula to a temporally guarded one can be decided and such a temporally guarded equivalent can be constructed automatically. In any case, such considerations rely on a formal semantics for $\mu\text{TL}_f \times K$ which we explain next.

Formulas of $\mu\text{TL}_f \times K$ are interpreted in pointed QTGs $(\mathfrak{G}, v, i)$. It is more convenient to fix a QTG $\mathfrak{G} = (G_0, \sigma_0, \dots, \sigma_{n-2}, G_{n-1})$ with $G_i = (S_i, \rightarrow_i, L_i)$ for $i = 0, \dots, n-1$ over the same set AP of atomic propositions, and then to define, by induction on the structure of a given formula φ , its *extension* $\llbracket \varphi \rrbracket_\rho^\mathfrak{G}$ in $\mathfrak{G}$, consisting of all points (v, i) such that $\mathfrak{G}, v, i \models \varphi$ in the same sense as for $\text{LTL}_f \times K$. Induction on the formula structure bears the burden of dealing with free variables in subformulas. This is done by extending the interpretation to a set $\llbracket \varphi \rrbracket_\rho^\mathfrak{G}$ of points (v, i) in time that satisfy φ under the assumption that any free variable X is satisfied by the points in $\rho(X)$. Thus, $\rho : \text{Var} \rightarrow \mathcal{P}(\bigcup_{i=0}^{n-1} S_i \times \{i\})$. We write $\rho[X \mapsto T]$ to denote the function which maps X to T and behaves like ρ otherwise.

$$\begin{aligned}
\llbracket p \rrbracket_\rho^\mathfrak{G} &= \{(v, i) \mid p \in L_i(v)\} \\
\llbracket X \rrbracket_\rho^\mathfrak{G} &= \rho(X) \\
\llbracket \neg \varphi \rrbracket_\rho^\mathfrak{G} &= \bigcup_{i=0}^{n-1} (S_i \times \{i\}) \setminus \llbracket \varphi \rrbracket_\rho^\mathfrak{G} \\
\llbracket \varphi \wedge \psi \rrbracket_\rho^\mathfrak{G} &= \llbracket \varphi \rrbracket_\rho^\mathfrak{G} \cap \llbracket \psi \rrbracket_\rho^\mathfrak{G} \\
\llbracket \odot \varphi \rrbracket_\rho^\mathfrak{G} &= \bigcup_{i=0}^{n-2} \{(v, i) \mid v \in S_i, \sigma_i(v) \text{ exists and } (\sigma_i(v), i+1) \in \llbracket \varphi \rrbracket_\rho^\mathfrak{G}\}
\end{aligned}$$

$$\begin{aligned} \llbracket \text{rec } X.\varphi \rrbracket_\rho^\mathfrak{G} &= \bigcap \left\{ T \subseteq \bigcup_{i=0}^{n-1} S_i \times \{i\} \mid \llbracket \varphi \rrbracket_{\rho[X \rightarrow T]}^\mathfrak{G} \subseteq T \right\} \\ \llbracket \Diamond \varphi \rrbracket_\rho^\mathfrak{G} &= \bigcup_{i=0}^{n-1} \{(v, i) \in S_i \times \{i\} \mid \text{there is } u \in S_i \text{ s.t. } v \rightarrow_i u \text{ and } (u, i) \in \llbracket \varphi \rrbracket_\rho^\mathfrak{G}\} \end{aligned}$$

It is standard to show that sentences φ are invariant under variable interpretations in the sense that $\llbracket \varphi \rrbracket_\rho^\mathfrak{G} = \llbracket \varphi \rrbracket_{\rho'}^\mathfrak{G}$ for any $\mathfrak{G}$, closed φ and ρ, ρ' . Hence, in such cases we can simply write $\llbracket \varphi \rrbracket^\mathfrak{G}$ or, likewise, $\mathfrak{G}, v, i \models \varphi$ for some v .

The notion of equivalence between two sentences, $\varphi \equiv \psi$, carries over from $\text{LTL}_f \times K$ to $\mu\text{TL}_f \times K$. Since sentences of the latter and formulas of the former are both interpreted in pointed QTGs, equivalence between these is also well-defined.

We get an embedding of $\text{LTL}_f \times K$ into $\mu\text{TL}_f \times K$ by a straight-forward extension of the known translation from LTL into μTL where $\varphi \cup \psi$ becomes $\text{rec } X.\psi \vee (\varphi \wedge \bigcirc X)$.

► **Lemma 2.** *For every $\varphi \in \text{LTL} \times K$ there is $\varphi' \in \mu\text{TL}_f \times K$ s.t. $|\varphi'| = \mathcal{O}(|\varphi|)$ and $\varphi' \equiv \varphi$.*

For reasoning about fixpoint formulas it is useful to introduce approximants. Since we are only interested in finite models, we do not need trans-finite ordinals in their construction. For a fixpoint formula $\text{rec } X.\varphi$ we define its approximants for all $i \in \mathbb{N}$ via $\text{rec}^0 X.\varphi := \text{ff}$ and $\text{rec}^{i+1} X.\varphi := \varphi[\text{rec}^i X.\varphi/X]$. The following enables reasoning about formulas with free variables by interpreting them as approximants. Its proof is standard, relies on temporal guardedness and is omitted.

► **Lemma 3.** *Let $\mathfrak{G}$ be a QTG of some length n . For every $\text{rec } X.\varphi \in \mu\text{TL}_f \times K$ and every variable assignment ρ we have $\llbracket \text{rec } X.\varphi \rrbracket_\rho^\mathfrak{G} = \llbracket \text{rec}^n X.\varphi \rrbracket_\rho^\mathfrak{G}$.*

3 Temporal Bisimulations

We adjust the well-known notion of bisimulation to the setting of QTGs.

► **Definition 4.** *Let $\mathfrak{G} = (G_0, \sigma_0, \dots, \sigma_{n-2}, G_{n-1})$ and $\mathfrak{G}' = (G'_0, \sigma'_0, \dots, \sigma'_{n'-2}, G'_{n'-1})$ be two QTGs, not necessarily of the same length, with $G_i = (S_i, \rightarrow_i, L_i)$ and $G'_j = (S'_j, \rightarrow'_j, L'_j)$ for $i = 0, \dots, n-1$ and $j = 0, \dots, n'-1$.*

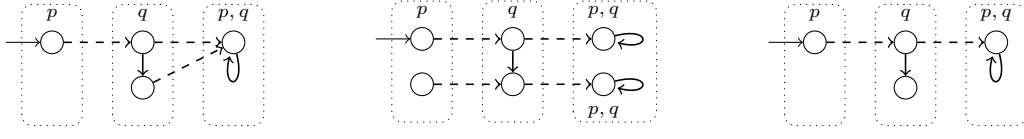
A relation β between points of $\mathfrak{G}$ and $\mathfrak{G}'$ is a temporal bisimulation if for all $(v, i) \beta (u, j)$ we have

- (1) $L_i(v) = L'_j(u)$,
 - (2) *for all $v' \in S_i$ with $v \rightarrow_i v'$ there is $u' \in S'_j$ with $u \rightarrow'_j u'$ and $(v', i) \beta (u', j)$,*
 - (3) *for all $u' \in S'_j$ with $u \rightarrow'_j u'$ there is $v' \in S_i$ with $v \rightarrow_i v'$ and $(v', i) \beta (u', j)$,*
 - (4) $\sigma_i(v)$ exists iff $\sigma'_j(u)$ exists, and in case they do, $(\sigma_i(v), i+1) \beta (\sigma'_j(u), j+1)$.
- As usual, two points (v, i) and (u, j) are temporally bisimilar, written $(\mathfrak{G}, v, i) \rightleftharpoons (\mathfrak{G}', u, j)$ if there is a bisimulation β between $\mathfrak{G}$ and $\mathfrak{G}'$ such that $(v, i) \beta (u, j)$.*

In condition (4), it should be understood that $\sigma_i(v)$ does not exist whenever $i = n-1$, because the function σ_{n-1} itself does not exist (and similarly for when $j = n'-1$).

In the following, we will simply speak of bisimulations etc. when in fact we mean temporal bisimulations. No confusion should arise, as ordinary bisimulations play no technical role here.

The following theorem states that $\mu\text{TL}_f \times K$ is invariant under temporal bisimulations. The proof is the same as for one-dimensional μ -calculi (see, for instance, [12]). Finiteness of QTGs in length simplifies the proof because fixpoints are reached after a finite amount of



■ **Figure 1** Examples of QTGs, resp. a TG (in the middle), concerning temporal bisimilarity.

iterations, cf. Lemma 3. We introduce elementary equivalence between two pointed QTGs in the usual way: we have $(\mathfrak{G}, v, i) \equiv_{\mu\text{TL}_f \times K} (\mathfrak{G}', u, j)$ if for all $\varphi \in \mu\text{TL}_f \times K$: $\mathfrak{G}, v, i \models \varphi$ iff $\mathfrak{G}', u, j \models \varphi$.

► **Theorem 5.** *For two pointed QTGs $(\mathfrak{G}, v, i)$ and $(\mathfrak{G}', u, j)$ we have $\mathfrak{G}, v, i \rightleftharpoons \mathfrak{G}', u, j$ iff $(\mathfrak{G}, v, i) \equiv_{\mu\text{TL}_f \times K} (\mathfrak{G}', u, j)$.*

The proof is in Appendix A. Whilst Thm. 5 is not surprising, it is significant for a study of the expressiveness of TGNNs that are interpreted over TGs. Note that, unlike the class of QTGs, the class of TGs is not closed under temporal bisimulations. Specifically, the two pointed QTGs shown on the left and in the middle of Fig. 1, with spatial neighbourhood relation shown in solid arrows and temporal successors in dashed arrows, are bisimilar but only one of them is a TG. These two suggest that QTGs can be “unwound” and given extra dummy nodes in early graphs of the sequence in order to turn them into a bisimilar TG. However, the QTG on the right show that this is not possible, specifically when there are maximal paths from the initial point that do not have the same number of temporal transitions in them.

One may argue that this calls for a stronger notion of quasi-temporal graph, namely sequences $(G_0, \sigma_0, \dots, \sigma_{n-2}, G_{n-1})$ such that all temporal successor functions σ are *total*. We call such QTGs (temporally) *synchronised*. The aforementioned technique of unwinding temporal paths and potentially completing their pasts yields the following result.

► **Theorem 6.** *For every pointed and temporally synchronised QTG $(\mathfrak{G}, v, 0)$ there is a pointed TG $(\mathfrak{G}', v, 0)$ with the same temporal length and at most polynomial blow-up in nodes such that $(\mathfrak{G}, v, 0) \rightleftharpoons (\mathfrak{G}', v, 0)$.*

Thm. 6 does not hold for temporally unsynchronised QTGs. This is also witnessed by the formula $p \wedge \bigcirc(q \wedge \Diamond \bigcirc \text{ff} \wedge \bigcirc \text{tt})$ which is satisfied by the QTG on the right in Fig. 1 but cannot be satisfied by a synchronised QTG.

4 The Satisfiability Problems

We study the computational complexity of the satisfiability problems for $\text{LTL}_f \times K$ and $\mu\text{TL}_f \times K$ over the class of QTGs. We begin with lower bounds.

4.1 Lower Bounds

The satisfiability problem for LTL and also LTL_f is well-known to be PSPACE-complete [32]. The lower bound trivially carries over to $\text{LTL}_f \times K$ since every formula of LTL_f is also one of $\text{LTL}_f \times K$ and a QTG model $(\mathfrak{G}, v, 0)$ of the latter is easily turned into a pure linear-time model of the form $v, \sigma_{0,1}(v), \sigma_{0,2}(v), \dots, \sigma_{0,n-1}(v)$ for the respective temporal successor functions $\sigma_0, \dots, \sigma_{n-2}$.

► **Theorem 7.** *The satisfiability problem for $\text{LTL}_f \times K$ is PSPACE-hard.*

In the pure linear-time setting, μTL has higher expressiveness than LTL but its satisfiability problem is not more difficult, at least not more than by a polynomial overhead: it is also PSPACE-complete [20]. Interestingly, $\mu\text{TL}_f \times \text{K}$ appears to be more difficult than $\text{LTL}_f \times \text{K}$. We show that its satisfiability problem is EXPTIME-hard. This is at least curious in that $\mu\text{TL}_f \times \text{K}$ appears to be a linear-time *temporal* logic whose satisfiability problems are typically complete for space-complexity classes. It is typically the satisfiability problems for branching-time temporal logics that tend to be complete for time-complexity classes [13, 15, 14, 36, 19] with the exception of K which is arguably not really a *temporal* logic as its formulas cannot “look” arbitrarily deep into a structure. The satisfiability problem for K is also known to be PSPACE-complete [23].

So the combination of the PSPACE-complete logics LTL_f and K into $\text{LTL}_f \times \text{K}$ turns out not to be more difficult than PSPACE – a matching upper bound is given below – whilst the combination of the PSPACE-complete logics μTL and K becomes EXPTIME-hard.

We show EXPTIME-hardness by a reduction from a tiling game problem, known to be EXPTIME-complete [10] as it captures the essence of the word problem for alternating, space bounded Turing machines and $\text{APSPACE} = \text{EXPTIME}$ [9].

► **Definition 8.** A tiling system is a $\mathcal{T} = (T, H, V, t_I, t_F)$ where T is a finite set of tiles with two designated initial and final tiles $t_I, t_F \in T$ such that $t_I \neq t_F$. $H, V \subseteq T \times T$ are the horizontal and vertical matching relation.

The n -corridor tiling game is played between players $\mathbf{S}$ and $\mathbf{C}$ who successively places tiles on the grid $\{0, \dots, n-1\} \times \mathbb{N}$ row-by-row, starting with row 0, i.e. placing tiles on the coordinates $(0, 0), \dots, (n-1, 0)$, then filling row 1, etc. The filling of the i -th row is done as follows.

1. Player $\mathbf{S}$ chooses a tile $t \in T$ and places it on the coordinate $(0, i)$.
 2. Player $\mathbf{C}$ chooses tiles $t_1, \dots, t_{n-1} \in T$ places them on the coordinates $(1, i), \dots, (n-1, i)$.
- This can also be seen as the successive construction of a mapping $\tau : [n] \times \mathbb{N} \rightarrow T$. Thus, we will write $\tau(j, i)$ for the tile placed in the j -th column in row i .

Player $\mathbf{C}$ wins a play constructing ρ if one of the following conditions holds.

- $\rho(0, 0) \neq t_I$, i.e. $\mathbf{S}$ did not choose the designated initial tile at the very beginning.
- There is $i \in \mathbb{N}$ such that $(\tau(0, i), \tau(0, i+1)) \notin V$, i.e. player $\mathbf{S}$ has violated the vertical matching condition when placing a tile.
- There are $j \in [n]$, $i \in \mathbb{N}$ such that $\tau(j, i) = t_F$, i.e. some player placed the final tile.

Likewise, $\mathbf{S}$ wins the play if there is $j \in \{0, \dots, n-2\}$ and $i \in \mathbb{N}$ such that $(\tau(j, i), \tau(j+1, i)) \notin H$, i.e. $\mathbf{C}$ placed a tile does not match its neighbour horizontally.

The n -corridor tiling game problem is the following. Given a tiling system $\mathcal{T}$ and an $n \in \mathbb{N}$ encoded unarily, decide whether player $\mathbf{C}$ has a winning strategy for the n -corridor tiling game.

► **Proposition 9** ([10]). The n -corridor tiling game problem is EXPTIME-complete.

► **Theorem 10.** Satisfiability for $\mu\text{TL}_f \times \text{K}$ over quasi-temporal graphs is EXPTIME-hard.

Proof. By a reduction from the n -corridor tiling game problem. Let $\mathcal{T} = (T, H, V, t_I, t_F)$ and $n \in \mathbb{N}$ be given, with n encoded unarily. This means that the construction only needs to be polynomial in the value of n , not necessarily the size needed to write down n succinctly in binary encoding for instance.

We construct a formula $\Phi_{\mathcal{T}, n} \in \mu\text{TL}_f \times \text{K}$ that is satisfied in a QTG iff $\mathbf{C}$ has a winning strategy for the game on $\mathcal{T}$ and n . Intuitively such a winning strategy can be seen as a tree with nodes representing tiled rows (and separating nodes in between) as shown in Fig. 2.

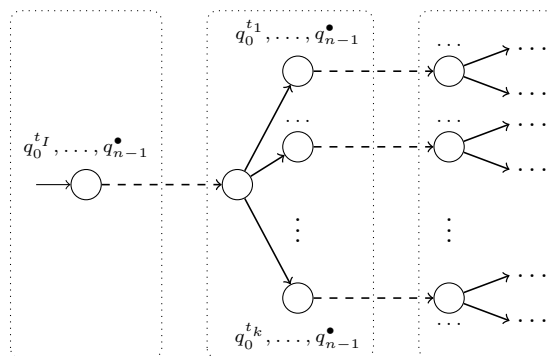


Figure 2 Representation of winning strategies in n -corridor tiling games as quasi-temporal graphs.

We use $\text{AP} = T \times [n]$ and write q_i^t for the proposition (t, i) . Note that an entirely tiled row $t_0, \dots, t_{n-1}$ can be represented in the label of a single graph node v as $L(v) = \{q_0^{t_0}, \dots, q_{n-1}^{t_{n-1}}\}$. Fig. 2 also shows how the tree-like winning strategy can be seen as a quasi-temporal graph in which every graph has a designated node that is the temporal successor of a node in the previous graph, and has multiple spatial neighbours. The branching represents the choices that $\mathbf{S}$ makes at the start of each row.

We construct $\Phi_{\mathcal{T},n}$ as the conjunction of the following five formulas, resp. properties.

(i) The labeling on the first node and every second one after that needs to represent the tiling of a row in the sense that for every $j \in [n]$ the label contains a unique proposition q_j^t .

$$\varphi_{\text{rows}} := \text{rec } X. \left(\bigwedge_{j=0}^{n-1} \bigvee_{t \in T} q_j^t \wedge \bigwedge_{\substack{t' \in T \\ t' \neq t}} \neg q_j^{t'} \right) \wedge \odot \square X$$

(II) The first tile is t_I . This is expressed simply as $\varphi_{\text{init}} := q_0^{t_I}$. (III) The horizontal matching relation is respected everywhere.

$$\varphi_{\text{horiz}} := \text{rec } X. \left(\bigwedge_{j=0}^{n-2} \bigwedge_{(t,t') \notin H} \neg(q_j^t \wedge q_{j+1}^{t'}) \right) \wedge \odot \square X$$

(IV) The vertical matching relation is respected everywhere that $\mathbf{C}$ makes a move.

$$\varphi_{\text{vert}} := \text{rec } X. \left(\bigwedge_{j=1}^{n-1} \bigwedge_{(t,t') \notin V} \neg(q_j^t \wedge \bigcirc \diamond q_j^{t'}) \right) \wedge \odot \square X$$

(v) The tree representing a winning strategy for **C** contains all the choices that are possible for **S**, unless the final tile has been placed already. We use the abbreviation tV for $\{t' \mid (t, t') \in V\}$.

$$\varphi_{\text{play}} := \text{rec } X. \left(\left(\bigwedge_{j=0}^{n-1} \neg q_j^{t_F} \right) \rightarrow \left(\bigwedge_{t \in T} q_0^t \rightarrow \bigwedge_{t' \in tV} \bigcirc \diamond q_0^{t'} \right) \right) \wedge \odot \square X$$

Note that, if $tV = \emptyset$ then the corresponding conjunction becomes true, and no diamonds need to be satisfied. This guarantees the existence of finite models representing winning strategies in which **C** wins plays because **S** has no legal move to play.

$$\begin{array}{ll}
(\text{Or}) \quad \frac{\varphi_i, \Phi}{\varphi_1 \vee \varphi_2, \Phi} & (\text{Mod}) \quad \frac{\Phi, \Psi \quad \varphi_1, \Gamma \quad \dots \quad \varphi_k, \Gamma}{\bigcirc \Phi, \odot \Psi, \diamond \varphi_1, \dots, \diamond \varphi_k, \square \Gamma, \Lambda} \quad \Phi \neq \emptyset \\
(\text{And}) \quad \frac{\varphi_1, \varphi_2, \Phi}{\varphi_1 \wedge \varphi_2, \Phi} & (\text{Mod}_0) \quad \frac{\varphi_1, \Gamma \quad \dots \quad \varphi_k, \Gamma}{\odot \Psi, \diamond \varphi_1, \dots, \diamond \varphi_k, \square \Gamma, \Lambda} \\
(\text{FP}) \quad \frac{\varphi, \Phi}{X, \Phi} \quad fp(X) = \text{rec } X.\varphi & (\text{Var}) \quad \frac{X, \Phi}{\text{rec } X.\varphi, \Phi}
\end{array}$$

■ **Figure 3** Rules for the satisfiability tableaux for $\mu\text{TL}_f \times \text{K}$.

Clearly, $\Phi_{\mathcal{T},n} := \varphi_{\text{rows}} \wedge \varphi_{\text{init}} \wedge \varphi_{\text{horiz}} \wedge \varphi_{\text{vert}} \wedge \varphi_{\text{play}}$ is of polynomial size in $|\mathcal{T}|$ and the value of n . Moreover, a quasi-temporal graph satisfying it represents a winning strategy in the way that is sketched above, and vice-versa. This construction therefore facilitates a polynomial reduction from the n -corridor tiling game problem to the satisfiability problem of $\mu\text{TL}_f \times \text{K}$. ◀

4.2 Upper Bounds

We begin with an upper bound for the more general logic $\mu\text{TL}_f \times \text{K}$. We characterise its satisfiability problem in terms of tableaux as witnesses for satisfiability and then show that their existence can be decided in exponential time.

We use Greek capital letters to denote sets of formulas. We avoid braces and simply write φ, Φ for the set $\{\varphi\} \cup \Phi$, additionally implying $\varphi \notin \Phi$. For the modal and temporal next-operators $\otimes \in \{\diamond, \square, \bigcirc, \odot\}$ we write $\otimes\Phi$ to denote a set of formulas in which everyone starts with the operator $\otimes$. In the context of such an $\otimes\Phi$, Φ then denotes the set of all formulas that are obtained from $\otimes\Phi$ by stripping of the top-level temporal, resp. modal operator $\otimes$. The symbol Λ is solely used for a propositionally consistent set of literals, i.e. a set of formulas of the form q or $\neg q$ that does not contain both the positive and negative version of any proposition.

► **Definition 11.** A tableau for an $\text{LTL}_f \times \text{K}$, resp. $\mu\text{TL}_f \times \text{K}$ formula φ in negation normal form is a finite tree whose nodes are labeled with sets of subformulas of φ , built according to the rules presented in Fig. 3. The tableau is closed if all its branches end in an axiom, i.e. a set of the form $\odot \Psi, \square \Gamma, \Lambda$ with Ψ and Γ possibly empty, and Λ being a consistent set of literals.

Remember that $\mu\text{TL}_f \times \text{K}$ formulas are required to be temporally guarded. Hence, any tableau for such a formula φ_0 is structured as follows. Starting with the root node there are a linearly (in $|\varphi_0|$) bounded number of applications of the rules (Or), (And), (FP) and (Var), leading to a non-branching growth of the tableau tree until a formula set is reached that matches the premise of either of the rules (Mod) or (Mod₀). We call such a sequence of formula sets a *block*. The application of either of the latter rules creates a branching of degree given by the number of $\diamond$ -formulas, i.e. also linear in $|\varphi_0|$. From the premises obtained after the application of either of the modal rules, new blocks can be obtained just like they are from the root node. Hence, any such tableau can be compressed into a tree whose nodes are the blocks and whose branching is given by the modal rules.

This tableau calculus is sound and complete w.r.t. satisfiability of $\mu\text{TL}_f \times \text{K}$ formulas over the class of QTGs.

$$\begin{array}{c}
(\mathbf{U}^-) \frac{\psi, \Phi}{\varphi \mathbf{U} \psi, \Phi} \qquad (\mathbf{U}^+) \frac{\varphi, \bigcirc(\varphi \mathbf{U} \psi), \Phi}{\varphi \mathbf{U} \psi, \Phi} \qquad (\mathbf{R}^-) \frac{\psi, \varphi, \Phi}{\varphi \mathbf{R} \psi, \Phi} \\
(\mathbf{R}^+) \frac{\psi, \bigcirc(\varphi \mathbf{R} \psi), \Phi}{\varphi \mathbf{R} \psi, \Phi}
\end{array}$$

■ **Figure 4** Rules for the satisfiability tableaux for $\text{LTL}_f \times K$ to replace **(FP)** and **(Var)**.

► **Theorem 12.** *Let φ_0 be a formula of $\mu\text{TL}_f \times K$. It is satisfiable over the class of quasi-temporal graphs iff there is a closed tableau for φ_0 .*

The proof is in Appendix A and follows standard arguments: a closed tableau can be collapsed into a tree-like QTG model of the formula, and a given model can be used to guide the construction of a closed tableau.

An immediate consequence of Thm. 12 is the fact that satisfiability in $\mu\text{TL}_f \times K$ over QTGs can be decided in exponential time.

► **Theorem 13.** *The satisfiability problem for $\mu\text{TL}_f \times K$ over QTGs is in EXPTIME.*

Proof. We sketch an alternating, polynomially space-bounded algorithm [9]. Note that every branch of length more than 2^{2^n} for formulas of size n must repeat some formula set. Hence, tableaux can be assumed to be of height at most 2^{2^n} . Then an alternating algorithm can check for the existence of such a “small” tableau by nondeterministically guessing applications of rule **(Or)** and deterministically applying **(And)**, **(FP)** and **(Var)** whenever needed. When eventually rule **(Mod)** or **(Mod₀)** need to be applied, the algorithm universally branches to follow all premises. Additionally, it maintains a counter that is started at 2^{2^n} and decreased in every step of a rule application. If the counter reaches 0, it rejects. It accepts when a branch can be closed.

The space needed for this is polynomial: it only needs to store a formula set and a counter of maximal value 2^{2^n} . The latter only requires polynomially many bits. The former is also only of polynomial size, as it only requires the storing of a single bit per subformula. ◀

So this raises the question after the exact complexity of $\text{LTL}_f \times K$. The crucial use of unbounded alternations between modal and temporal operators in the fixpoint formulas constructed to simulate the tiling game problem would suggest that satisfiability for $\text{LTL}_f \times K$ is not as hard as that for $\mu\text{TL}_f \times K$. The fact that LTL_f and μTL_f have the same asymptotic complexity may suggest that so should $\text{LTL}_f \times K$ and $\mu\text{TL}_f \times K$. This is not true, though. The lack of a general fixpoint operator that allows formulas to look arbitrarily deep into graphs (that are sufficiently far away in a QTG) makes for a simpler complexity, at least up to common complexity-theoretic assumptions.

To formally reason about satisfiability in $\text{LTL}_f \times K$ we introduce a similar tableau calculus: simply replace rules **(FP)** and **(Var)** by the four rules shown in Fig. 4. Correctness of these tableau is proved in the same way as in Thm. 12. The only difference here is that the formulas occurring in a tableau for φ_0 are not just subformulas of φ_0 . Instead, for each subformula of the form $\varphi \mathbf{U} \psi$ or $\varphi \mathbf{R} \psi$, the extended subformulas $\bigcirc(\varphi \mathbf{U} \psi)$ and $\bigcirc(\varphi \mathbf{R} \psi)$ may also occur. This is asymptotically harmless. Moreover, no additional reasoning for free variables is needed in the correctness proof.

► **Corollary 14.** *Let φ_0 be a formula of $\text{LTL}_f \times K$. It is satisfiable over the class of quasi-temporal graphs iff there is a closed tableau for φ_0 .*

The crucial insight leading to an improved upper complexity bound is the observation that for $\text{LTL}_f \times K$ a nondeterministic, rather than alternating polynomial-space algorithm suffices. Then by Savitch's Theorem [31] we get a PSPACE upper bound. This may seem strange, though, because rule (Mod) and (Mod₀) still induce genuine branching in the tableaux, and nondeterministic choices also need to be made in rule (0r) and the rules for temporal operators. However, in an $\text{LTL}_f \times K$ formula, the argument of a modal operator is genuinely smaller in the sense that in an application of rule (Mod) or (Mod₀) to a premise containing φ one cannot “go back” to $\Diamond \varphi$ or $\Box \varphi$, at least not through φ .

► **Theorem 15.** *The satisfiability problem for $\text{LTL}_f \times K$ over QTGs is in PSPACE.*

Proof. An algorithm can nondeterministically check for satisfiability by successively applying rules whilst performing guesses for (0r), (U^-), (U^+), (R^-) and (R^+). No guess is needed for rule (And). When a formula set is reached that forms the end of a block, i.e. that requires an application of rule (Mod) or (Mod₀), we proceed as follows. Consider the case of (Mod), as the one for (Mod₀) is just a special case of that one. Suppose the number of premises for this rule application is k . Note that necessarily $k \leq n + 1$. The algorithm stores the $k - 1$ left premises and continues searching for a tableau for the rightmost premise. Upon successful termination it backtracks and continues with the next rightmost premise that has not been explored yet, re-using the space that was needed for the exploration of the previous premise.

Since the input formula is from $\text{LTL}_f \times K$, the recursion depth of this backtracking procedure is bounded: all but the leftmost premise of rule (Mod) have a modal depth that is strictly smaller than the modal depth of the conclusion. Hence, the call stack for the recursion in this algorithm has at most linear depth, and each stack frame has at most quadratic size, yielding an at most cubic space consumption. ◀

5 Conclusion

This paper studied the computational complexity of satisfiability for the two temporal-spatial logics $\text{LTL}_f \times K$ and $\mu\text{TL}_f \times K$, interpreted over quasi-temporal graphs. The results draw a slightly unexpected picture: while satisfiability for the underlying one-dimensional logics LTL_f and μTL_f is PSPACE-complete in either case, the two-dimensional combinations behave differently. $\text{LTL}_f \times K$ remains PSPACE-complete, whereas $\mu\text{TL}_f \times K$ becomes EXPTIME-complete, a complexity that is more reminiscent of branching-time temporal logics than of linear-time ones.

The use of QTGs rather than the more restrictive class of TGs is justified by the temporal-spatial bisimulation invariance of $\mu\text{TL}_f \times K$ established in Thm. 5. The class of TGs is not closed under bisimulation, which from a logical perspective makes QTGs the more natural class to consider. Nevertheless, satisfiability over the class of TGs is a meaningful problem in its own right, not least because TGs are the objects TGNNs reason about. TG-satisfiability is a strictly stronger condition than QTG-satisfiability as seen with a formula like $\Diamond \odot \text{ff} \wedge \bigcirc \text{tt}$ which is QTG-satisfiable, but neither satisfiable over synchronised QTGs nor TGs. Hence, the decision procedures do not directly transfer to the TG-setting. However, the lower bounds do. PSPACE-hardness for $\text{LTL}_f \times K$ and EXPTIME-hardness for $\mu\text{TL}_f \times K$ remain valid over the class of TGs. For matching upper bounds, the situation is more delicate. The tableau construction in Thm. 12 yields a QTG whose temporal successor functions are neither total nor injective. Thm. 6 provides a partial bridge by showing that every synchronised QTG can be unfolded into a bisimilar TG, but the formula above witnesses that this bridge cannot be extended to unsynchronised QTGs. We conjecture that TG-satisfiability is no harder than

QTG-satisfiability for either logic, with the upper bound obtained by an adaptation of the tableau calculus that enforces totality and injectivity of the temporal successor functions on closed branches. A precise proof of this, and the question of whether the formulation can be carried out without further blow-up in complexity, is left for future work.

The original motivation for studying these logics stems from their role as yardsticks for the expressive power of Temporal Graph Neural Networks. A potential next step is to replace the spatial logic K by stronger modal logics, such as $K^\#$ [1] or logics with Presburger-style counting [4], which are often considered in the expressiveness research of TGNNs [29, 5].

References

- 1 C. Areces, G. Hoffmann, and A. Denis. Modal logics with counting. In *Proc. 17th Workshop on Logic, Language, Information and Computation, WoLLIC'10*, volume 6188 of *LNCS*, pages 98–109. Springer, 2010. doi:10.1007/978-3-642-13824-9_9.
- 2 B. Banieqbal and H. Barringer. Temporal logic with fixed points. In *Proc. Coll. on Temporal Logic in Specification*, volume 398 of *LNCS*, pages 62–73. Springer, 1989. doi:10.1007/3-540-51803-7_22.
- 3 P. Barceló, E. V. Kostylev, M. Monet, J. Pérez, J. L. Reutter, and J. P. Silva. The logical expressiveness of graph neural networks. In *Proc. 8th Int. Conf. on Learning Representations, ICLR'20*. OpenReview.net, 2020. URL: <https://openreview.net/forum?id=r1lZ7AEKvB>.
- 4 B. Bednarczyk, M. Orlowska, A. Pacanowska, and T. Tan. On classical decidable logics extended with percentage quantifiers and arithmetics. In *Proc. 41st IARCS Annual Conf. on Foundations of Software Technology and Theoretical Computer Science, FSTTCS'21*, volume 213 of *LIPIcs*, pages 36:1–36:15. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2021. doi:10.4230/LIPIcs.FSTTCS.2021.36.
- 5 M. Benedikt, C.-H. Lu, B. Motik, and T. Tan. Decidability of graph neural networks via logical characterizations. In *Proc. 51st Int. Coll. on Automata, Languages, and Programming, ICALP'24*, volume 297 of *LIPIcs*, pages 127:1–127:20. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2024. doi:10.4230/LIPIcs.ICALP.2024.127.
- 6 J. Bradfield and C. Stirling. Modal mu-calculi. In P. Blackburn, J. van Benthem, and F. Wolter, editors, *Handbook of Modal Logic: Studies in Logic and Practical Reasoning Volume 3*, volume 3 of *Studies in logic and practical reasoning*, pages 721–756. North-Holland, 2007. doi:10.1016/S1570-2464(07)80015-2.
- 7 J. Bradfield and I. Walukiewicz. The mu-calculus and model checking. In *Handbook of Model Checking*, pages 871–919. Springer, 2018. doi:10.1007/978-3-319-10575-8_26.
- 8 F. Bruse, O. Friedmann, and M. Lange. On guarded transformation in the μ -calculus. *Logic Journal of the IGPL*, 23(2):194–216, 2015. doi:10.1093/jigpal/jzu030.
- 9 A. K. Chandra, D. C. Kozen, and L. J. Stockmeyer. Alternation. *J. ACM*, 28(1):114–133, 1981. doi:10.1145/322234.322243.
- 10 B. S. Chlebus. Domino-tiling games. *Journal of Computer and System Sciences*, 32(3):374–392, 1986. doi:10.1016/0022-0000(86)90036-X.
- 11 M. Dam. Fixpoints of Büchi automata. In *Proc. 12th Conf. on Foundations of Software Technology and Theoretical Computer Science, FST&TCS'92*, volume 652 of *LNCS*, pages 39–50. Springer, 1992. doi:10.1007/3-540-56287-7_93.
- 12 S. Demri, V. Goranko, and M. Lange. *Temporal Logics in Computer Science: Finite-State Systems*. Cambridge Tracts in Theoretical Computer Science. Cambridge University Press, 2016. doi:10.1017/CB09781139236119.
- 13 E. A. Emerson and J. Y. Halpern. Decision procedures and expressiveness in the temporal logic of branching time. *J. Comput. Syst. Sci.*, 30(1):1–24, 1985. doi:10.1016/0022-0000(85)90001-7.
- 14 E. A. Emerson and A. P. Sistla. Deciding full branching time logic. *Inf. Control.*, 61(3):175–201, 1984. doi:10.1016/S0019-9958(84)80047-9.
- 15 M. J. Fischer and R. E. Ladner. Propositional dynamic logic of regular programs. *J. Comput. Syst. Sci.*, 18(2):194–211, 1979. doi:10.1016/0022-0000(79)90046-1.

- 16 D. M. Gabbay, A. Kurucz, F. Wolter, and M. Zakharyashev. *Many-dimensional modal logics: theory and applications*. Studies in Logic and the Foundations of Mathematics. Elsevier, 2003.
- 17 G. De Giacomo and M. Y. Vardi. Linear temporal logic and linear dynamic logic on finite traces. In *Proc. 23rd Int. Joint Conf. on A.I., IJCAI'13*, pages 854–860. IJCAI/AAAI, 2013. URL: <http://www.aaai.org/ocs/index.php/IJCAI/IJCAI13/paper/view/6997>.
- 18 J. Gilmer, S. S. Schoenholz, P. F. Riley, O. Vinyals, and G. E. Dahl. Neural message passing for quantum chemistry. In *Proc. 34th Int. Conf. on Machine Learning, ICML'17*, volume 70 of *Proc. Machine Learning Research*, pages 1263–1272. PMLR, 2017. URL: <http://proceedings.mlr.press/v70/gilmer17a.html>.
- 19 J. Johannsen and M. Lange. CTL⁺ is complete for double exponential time. In *Proc. 30th Int. Coll. on Automata, Logics and Programming, ICALP'03*, volume 2719 of *LNCS*, pages 767–775. Springer, 2003. doi:10.1007/3-540-45061-0_60.
- 20 R. Kaivola. A simple decision method for the linear time μ -calculus. In *Proc. Int. Workshop on Structures in Concurrency Theory, STRICT'95*, pages 190–204, 1995. doi:10.1007/978-1-4471-3078-9_13.
- 21 B. Knaster. Un théorème sur les fonctions d'ensembles. *Annals Soc. Pol. Math.*, 6:133–134, 1928.
- 22 D. Kozen. Results on the propositional μ -calculus. *Theor. Comput. Sci.*, 27:333–354, 1983. doi:10.1016/0304-3975(82)90125-6.
- 23 R. E. Ladner. The computational complexity of provability in systems of modal propositional logic. *SIAM J. Comput.*, 6(3):467–480, 1977. doi:10.1137/0206033.
- 24 Y. Liu, Z. Duan, C. Tian, and B. Cui. Satisfiability of linear time μ -calculus on finite traces. In *Proc. 22nd Int. Conf. on Computing and Combinatorics, COCOON'16*, volume 9797 of *LNCS*, pages 611–622. Springer, 2016. doi:10.1007/978-3-319-42634-1_49.
- 25 A. Longa, V. Lachi, G. Santin, M. Bianchini, B. Lepri, P. Lio, F. Scarselli, and A. Passerini. Graph neural networks for temporal graphs: State of the art, open challenges, and opportunities. *Trans. Mach. Learn. Res.*, 2023. doi:10.48550/arXiv.2302.01018.
- 26 C. Lutz, F. Wolter, and M. Zakharyashev. Temporal description logics: A survey. In *Proc. 15th Int. Symp. on Temporal Representation and Reasoning, TIME'08*, pages 3–14. IEEE, 2008. doi:10.1109/TIME.2008.14.
- 27 M. Marx and Y. Venema. *Multi-Dimensional Modal Logic*, pages 1–9. Springer, 1997. doi:10.1007/978-94-011-5694-3_1.
- 28 R. Mateescu. Local model-checking of modal μ -calculus on acyclic labeled transition systems. In *Proc. 8th Int. Conf. on Tools and Algorithms for the Construction and Analysis of Systems, TACAS'02*, volume 2280 of *LNCS*, pages 281–295. Springer, 2002. doi:10.1007/3-540-46002-0_20.
- 29 P. Nunn, M. Sälzer, F. Schwarzentruher, and N. Troquard. A logic for reasoning about aggregate-combine graph neural networks. In *Proc. 33rd Int. Joint Conf. on A.I., IJCAI'24*, pages 3532–3540. ijcai.org, 2024. URL: <https://www.ijcai.org/proceedings/2024/391>.
- 30 M. Sälzer, P. A. Wałęga, and M. Lange. The logical expressiveness of temporal gnns via two-dimensional product logics. In *Proc. 39th Annual Conf. on Neural Information Processing Systems, NeurIPS'25*. Openreview.net, 2025. doi:10.48550/arXiv.2505.11930.
- 31 W. J. Savitch. Relationships between nondeterministic and deterministic tape complexities. *J. Comput. Syst. Sci.*, 4(2):177–192, 1970. doi:10.1016/S0022-0000(70)80006-X.
- 32 A. P. Sistla and E. M. Clarke. The complexity of propositional linear temporal logics. *J. ACM*, 32(3):733–749, 1985. doi:10.1145/3828.3837.
- 33 J. Skarding, B. Gabrys, and K. Musiał. Foundations and modeling of dynamic networks using dynamic graph neural networks: A survey. *IEEE Access*, 9:79143–79168, 2021. doi:10.1109/ACCESS.2021.3082932.
- 34 A. Tarski. A lattice-theoretical fixpoint theorem and its application. *Pacific Journal of Mathematics*, 5:285–309, 1955. doi:10.2140/pjm.1955.5.285.

- 35 M. Y. Vardi. A temporal fixpoint calculus. In *Proc. Conf. on Principles of Programming Languages, POPL'88*, pages 250–259. ACM, 1988. doi:10.1145/73560.73582.
- 36 M. Y. Vardi and L. Stockmeyer. Improved upper and lower bounds for modal logics of programs. In *Proc. 17th Symp. on Theory of Computing, STOC'85*, pages 240–251. ACM, 1985. doi:10.1145/22145.22173.
- 37 I. Walukiewicz. Completeness of Kozen's axiomatisation of the propositional μ -calculus. *Inf. and Comput.*, 157(1-2):142–182, 2000. doi:10.1006/inco.1999.2836.

A Proof Details

Proof of Thm. 5. Let $\mathfrak{G} = (G_0, \sigma_0, \dots, \sigma_{n-2}, G_{n-1})$ and $\mathfrak{G}' = (G'_0, \sigma'_0, \dots, \sigma'_{n'-2}, G'_{n'-1})$ with $G_i = (S_i, \rightarrow_i, L_i)$ and $G'_j = (S'_j, \rightarrow'_j, L'_j)$.

“ $\Rightarrow$ ” By a slight abuse of notation, we write $\rightleftharpoons$ for the restriction of the bisimilarity relation to points in $\mathfrak{G}$ and $\mathfrak{G}'$. It is standard to argue that $\rightleftharpoons$ is the union of all bisimulation relations between $\mathfrak{G}$ and $\mathfrak{G}'$, and that it is itself a bisimulation between $\mathfrak{G}$ and $\mathfrak{G}'$.

Let $T \subseteq \bigcup_{i=0}^{n-1} S_i \times \{i\}$ and $T' \subseteq \bigcup_{j=0}^{n'-1} S'_j \times \{j\}$. We say that the pair (T, T') respects bisimilarity if for all $(v, i) \rightleftharpoons (u, j)$ we have $(v, i) \in T$ iff $(u, j) \in T'$. Hence, our goal is to show that for all $\varphi \in \mu\text{TL}_f \times \mathbf{K}$, the pair $(\llbracket \varphi \rrbracket^{\mathfrak{G}}, \llbracket \varphi \rrbracket^{\mathfrak{G}'})$ respects bisimilarity. In order to enable a proof by induction on the structure of formulas, we need to strengthen the claim to cover free variables. We say that a pair (ρ, ρ') of variable assignments for $\mathfrak{G}$ and $\mathfrak{G}'$ respectively respects bisimilarity if $(\rho(X), \rho'(X))$ respects bisimilarity for all $X \in \text{Var}$. We show that for all $\varphi \in \mu\text{TL}_f \times \mathbf{K}$ and all (ρ, ρ') that respect bisimilarity, we have that $(\llbracket \varphi \rrbracket_{\rho}^{\mathfrak{G}}, \llbracket \varphi \rrbracket_{\rho'}^{\mathfrak{G}'})$ respects bisimilarity.

The case $\varphi = p \in \text{AP}$ follows from condition (1) in Def. 4, and the case of $\varphi = X \in \text{Var}$ from the assumption about ρ and ρ' . The cases of the boolean operators $\wedge$ and $\neg$ follow directly from the hypothesis.

Let $\varphi = \bigcirc \psi$. Suppose that $(v, i) \rightleftharpoons (u, j)$ and $(v, i) \in \llbracket \bigcirc \psi \rrbracket_{\rho}^{\mathfrak{G}}$. Then $i < n - 1$, $\sigma_i(v)$ exists and $(\sigma_i(v), i + 1) \in \llbracket \psi \rrbracket_{\rho}^{\mathfrak{G}}$. By condition (4) in Def. 4 we have $j < n' - 1$, $\sigma'_j(u)$ exists and $(\sigma_i(v), i + 1) \rightleftharpoons (\sigma'_j(u), j + 1)$. By the induction hypothesis, $(\sigma'_j(u), j + 1) \in \llbracket \psi \rrbracket_{\rho'}^{\mathfrak{G}'}$ and thus $(u, j) \in \llbracket \bigcirc \psi \rrbracket_{\rho'}^{\mathfrak{G}'}$. The converse implication is treated symmetrically. Hence, $(\llbracket \varphi \rrbracket_{\rho}^{\mathfrak{G}}, \llbracket \varphi \rrbracket_{\rho'}^{\mathfrak{G}'})$ respects bisimilarity.

The case $\varphi = \Diamond \psi$ is treated similarly, using conditions (2) and (3) of Def. 4.

Finally, let $\varphi = \text{rec } X.\psi$. Suppose that $(v, i) \rightleftharpoons (u, j)$ and $(v, i) \in \llbracket \text{rec } X.\psi \rrbracket_{\rho}^{\mathfrak{G}}$. According to Lemma 3, we have $(v, i) \in \llbracket \text{rec}^n X.\psi \rrbracket_{\rho}^{\mathfrak{G}}$. The crucial insight here is that, while $\text{rec}^n X.\psi$ is generally larger than $\text{rec } X.\psi$ and not a subformula of it, $\text{rec}^n X.\psi$ does not contain X anymore. By a separate induction on the approximant level n we can infer that $(u, j) \in \llbracket \text{rec}^{n'} X.\psi \rrbracket_{\rho'}^{\mathfrak{G}'}$, making use of the main induction hypothesis for formulas not containing X anymore. A second application of Lemma 3 then yields $(u, j) \in \llbracket \text{rec } X.\psi \rrbracket_{\rho'}^{\mathfrak{G}'}$.

“ $\Leftarrow$ ” It suffices to show that $\equiv_{\mu\text{TL}_f \times \mathbf{K}}$ is a temporal bisimulation. Condition (1) is satisfied because propositions are formulas. Hence, suppose that the pair of points (v, i) and (u, j) could not be included because $L_i(v) \neq L'_j(u)$ then there is some $p \in \text{AP}$ such that $\mathfrak{G}, v, i \models p$ and $\mathfrak{G}', u, j \not\models p$ or vice-versa.

The reasoning for conditions (2) and (3) is the same as that for pure modal logic. Suppose that, say, (v, i) has a spatial successor (v', i) that cannot be matched by any successor (u', j) of (u, j) . Due to the co-inductive nature of bisimilarity, we can assume that there is a formula φ satisfied by (v', i) but not by any successor (u', j) of (u, j) . Then $\mathfrak{G}, v, i \models \Diamond \varphi$ but $\mathfrak{G}', u, j \not\models \Diamond \varphi$.

The reasoning for condition (4) is also done along the same line. Here the separating formula is of the form $\bigcirc \varphi$. ◀

Proof of Thm 12. “ $\Rightarrow$ ” Suppose that $(\mathfrak{G}, v_0, 0) \models \varphi_0$ for some QTG $\mathfrak{G} = (G_0, \sigma_0, \dots, \sigma_{n-2}, G_{n-1})$ with $G_i = (S_i, \rightarrow_i, L_i)$ and node $v \in S_0$.

We build a tableau for φ_0 , guided by nodes in $\mathfrak{G}$, starting with v_0 and the root node $\{\varphi_0\}$ of the tableau. Note that, by assumption, we have $(\mathfrak{G}, v_0, 0) \models \varphi_0$. The construction of the tableau will maintain the invariant that $(\mathfrak{G}, v, i) \models \varphi$ for all $\varphi \in \Phi$, provided that v is the current guide node, $v \in S_i$ and Φ is the current formula set at the end of a non-closed tableau branch. We call i the *height* of the tableau node.

Suppose that v is the current node, and the current node set Φ is not of the form $\bigcirc \Phi, \odot \Psi, \diamond \Delta, \square \Gamma, \Lambda$. Then it must contain a conjunction $\psi_1 \wedge \psi_2$, a fixpoint formula $\text{rec } X.\psi$, a variable X or a disjunction $\psi_1 \vee \psi_2$. In the first three cases we simply apply rule (And), (FP) or (Var) to deterministically obtain a new formula set that is the premise of the rule application. The current node remains to be v . It is easy to see that the invariant is maintained under rule (And).

For rules (FP) and (Var) this is less obvious because now the formula set may contain free variables. We refine the invariant as follows, thus also maintaining it with these two rules: if the current guide node is $v \in S_i$ and the current formula set is Φ then we have $(\mathfrak{G}, v, i) \models \varphi$ for all $\varphi \in \Phi$ where we interpret any variable $X \in \text{Var}$ as $\text{rec}^{n-i+1} X.\psi$ if $\text{fp}_{\varphi_0}(X) = \text{rec } X.\psi$.

At last, if Φ contains a disjunction $\psi_1 \vee \psi_2$ then by the invariant we have $(\mathfrak{G}, v, i) \models \psi_1 \vee \psi_2$. Hence, it is possible to choose one of the two disjuncts with rule (Or) to maintain the invariant.

Now suppose that the current node set is of the form $\bigcirc \Phi, \odot \Psi, \diamond \Delta, \square \Gamma, \Lambda$. Since $(\mathfrak{G}, v, i) \models \varphi$ for all φ in it, the following must be true.

- Λ must indeed be propositionally consistent because $q \in L_i(v)$ and $q \notin L_i(v)$ is impossible.
- If $\Phi \neq \emptyset$ then $i < n-1$ and $\sigma_i(v) \in S_{i+1}$, and for all $\varphi \in \Phi \cup \Psi$ we have $(\mathfrak{G}, \sigma_i(v), i+1) \models \varphi$.
- If $\Delta \neq \emptyset$ then for every $\varphi \in \Delta$ there is $u_\varphi \in S_i$ with $v \rightarrow_i u_\varphi$ and $(\mathfrak{G}, u_\varphi, i) \models \psi$ for all $\psi \in \{\varphi\} \cup \Gamma$.

This shows that rule (Mod) can be applied maintaining the invariant for new guide nodes $\sigma_i(v)$ and u_φ for $\varphi \in \Delta$. The reasoning for rule (Mod₀) is the same; just the second condition of the three becomes void.

This shows that, for as long as a tableau branch is not closed, the construction can be continued, guided by nodes in $\mathfrak{G}$. Temporal guardedness ensures that no branch of the tableau can have two applications of rule (Var) for the same variable without going through the leftmost premise of rule (Mod). On the other hand, (Var) is the only rule that increases the current formula set. Hence, the heights of nodes on every tableau branch must eventually be strictly increasing, but no node can have height $\geq n$. Thus, each branch must eventually be guided by a node with no temporal successor, closing the branch.

“ $\Leftarrow$ ” Suppose that φ_0 has a closed tableau. According to the observation made above, it can be collapsed into a tree of blocks. This tree can be rearranged as a quasi-temporal graph $\mathfrak{G} = (G_0, \sigma_0, G_1, \dots, G_n)$ by assigning each block as a node to graph G_i if the path from that block to the root block of the tableau contains i many steps through the leftmost premise of (Mod). The other connections between blocks obtained through non-leftmost premises of (Mod) or applications of (Mod₀) are translated into edges in the respective graphs. The temporal successors functions σ_i map any block to the block (necessarily in the next graph) that is started by the leftmost premise of the application of (Mod) at the end of the block.

One can then show, for $i = n, \dots, 0$ and every node v from G_i that is obtained as a collapsed block $\Phi_1, \dots, \Phi_k$ that $(\mathfrak{G}, v, i) \models \varphi$ for all $\varphi \in \Phi_1 \cup \dots \cup \Phi_k$. Again, we interpret a variable X occurring in a block forming a node in G_i as $\text{rec}^{n-i+1} X.\varphi$ when $\text{fp}_{\varphi_0}(X) = \text{rec } X.\varphi$. Consequently, since φ_0 is contained in the root block of the tableau, it has a model. ◀