Materialisation-Based Reasoning in DatalogMTL with Bounded Intervals

Przemysław A. Wałęga, Michał Zawidzki, Dingmin Wang, Bernardo Cuenca Grau

Department of Computer Science, University of Oxford, UK
{przemyslaw.walega, michal.zawidzki, dingmin.wang, bernardo.cuenca.grau}@cs.ox.ac.uk

Abstract
DatalogMTL is a powerful extension of Datalog with operators from metric temporal logic (MTL) which has received significant attention in recent years. In this paper, we investigate materialisation-based reasoning (a.k.a. forward chaining) in the context of DatalogMTL programs and datasets with bounded intervals, where partial representations of the canonical model are obtained through successive rounds of rule applications. Although materialisation does not naturally terminate in this setting, it is known that the structure of canonical models is ultimately periodic. Our first contribution in this paper is a detailed analysis of the periodic structure of canonical models; in particular, we formulate saturation conditions whose satisfaction by a partial materialisation implies the ability to recover the full canonical model via unfolding; this allows us to compute the actual periods describing the repeating parts of the canonical model as well as to establish concrete bounds on the number of rounds of rule applications required to achieve saturation. Based on these theoretical results, we propose a practical reasoning algorithm where saturation can be efficiently detected as materialisation progresses, and where the relevant periods used to evaluate entailment of queries via unfolding are efficiently computed. We have implemented our algorithm and our experiments suggest that our approach is both scalable and robust.

Introduction
DatalogMTL is a powerful rule-based language for representing temporal knowledge that has found applications in ontology-based data access (Brandt et al. 2018; Kikot et al. 2018; Kalayci et al. 2018; Koopmann 2019) and stream reasoning (Wałęga, Kaminski, and Cuenca Grau 2019). It extends Datalog (Ceri, Gottlob, and Tanca 1989) with operators from metric temporal logic (Koymans 1990) interpreted over the rational timeline. For example, the following DatalogMTL rule states that a person can travel if they had a negative COVID test at some point in the last 2 days (\(\neg \text{NegTest}\)) and have been double-vaccinated for 14 days (\([0,14]\)):

\[\text{CanTravel}(x) \leftarrow \neg [0,2] \text{NegTest}(x) \land [0,14] \text{DblVacc}(x)\]

Reasoning in DatalogMTL is of high complexity, which makes its adoption in applications problematic. In particular, satisfiability and fact entailment are ExpSpace-complete (Brandt et al. 2018) and PSpace-complete in data complexity (Wałęga et al. 2019). Decision procedures are based on the observation that, although the timeline is densely ordered, the canonical model of a program and a dataset can be partitioned according to discrete sequences of intervals, inside which all time points satisfy the same facts. This enables the reduction (with exponential blow-up) of fact entailment in DatalogMTL to satisfiability of linear temporal logic (LTL) formulas (Brandt et al. 2018) or to non-emptiness of Büchi automata (Wałęga et al. 2019). As a result, canonical models have a periodic structure, with periods of bounded length (Artale et al. 2021a). Although these techniques are useful for providing complexity upper bounds for reasoning, they are based on exponential reductions and were not designed for efficient implementation.

The most common technique of choice in scalable Datalog reasoners is materialisation (a.k.a. forward chaining), where facts entailed by a program and dataset are derived in successive rounds of rule applications until a fixpoint is reached (Bry et al. 2007; Motik et al. 2014; Carral et al. 2019; Bellomarini, Sallinger, and Gottlob 2018).\(^1\) Once the materialisation has been computed, queries can be answered directly and rules are not further considered. In contrast to Datalog where materialisation naturally terminates, in DatalogMTL a fixpoint may only be reachable after infinitely many rounds of rule applications. To circumvent this challenge, the MeTeoR reasoner (Wang et al. 2022) combines materialisation and automata-based reasoning, where automata construction ensures completeness where materialisation does not terminate; experiments, however, show a significant performance reduction in such cases (Wang et al. 2022). An alternative approach is to focus on DatalogMTL fragments for which materialisation is guaranteed to terminate (Wałęga, Zawidzki, and Cuenca Grau 2021); such fragments, however, impose significant restrictions effectively disallowing ‘recursion through time’.

In this paper, we investigate the design of reasoning algorithms for DatalogMTL that rely solely on materialisation, dispense with automata construction altogether, and do not limit recursion. We focus on bounded DatalogMTL programs and datasets, where \(\infty\) is not mentioned as an end-

\(^1\)In this setting, both the reasoning process and its output are often referred to as materialisation.
point of any interval; this is a natural and expressive fragment, in which reasoning is as hard as in the unrestricted language (Walęga, Zawidzki, and Cuenca Grau 2021).

Our first contribution is a detailed analysis of the periodic structure of canonical models. We formulate saturation conditions that a partial materialisation needs to satisfy so that the canonical model can be recovered via unfolding; this allows us to compute the actual periods describing the repeating parts of the model based only on the form of the partial materialisation constructed so far, as well as to establish concrete bounds on the number of rounds of rule applications required to achieve saturation. This is a challenging problem since DatalogMTL rules may recursively propagate facts towards both future and past time points, and hence regularities observed in a partial materialisation may not correspond to the periodic structure of the full canonical model.

Based on these theoretical results, we propose a practical reasoning algorithm where saturation can be efficiently detected as materialisation progresses, and where the relevant periods used to evaluate entailment queries via unfolding are efficiently computed. We have implemented our algorithm and evaluated its performance on a temporal extension of Lehigh University Benchmark (Wang et al. 2022) as well as using the iTemporal benchmark generator (Bellomarini, Nissl, and Sellinger 2022). Our results show that the overhead introduced by saturation checks during materialisation is negligible, saturation can be reached after a reasonable number of rounds of rule applications, and entailment checks performed after reaching saturation can be conducted very efficiently.

Preliminaries

We recapitulate the syntax and semantics of DatalogMTL, focusing on the standard continuous semantics over the rational timeline (Brandt et al. 2018; Walęga et al. 2019).

Time and Intervals. The (rational) timeline is the ordered set \( \mathbb{Q} \) of rational numbers; each element of the timeline is called a time point. We consider binary representations of integers, and represent each rational number as a fraction with an integer numerator and a positive integer denominator.

An interval is a non-empty subset of \( \mathbb{Q} \) satisfying two properties: (i) for all \( t_1, t_2, t_3 \in \mathbb{Q} \) with \( t_1 < t_2 < t_3 \) and \( t_1, t_3 \in \varrho \), it is the case that \( t_2 \in \varrho \), and (ii) both the greatest lower bound \( \varrho^- \) and the upper bound \( \varrho^+ \) of \( \varrho \) are in \( \mathbb{Q} \cup \{ -\infty, \infty \} \). The bounds \( \varrho^- \) and \( \varrho^+ \) are the left and right endpoints of \( \varrho \), respectively, and \( |\varrho| = \varrho^+ - \varrho^- \) is the length of \( \varrho \). Interval \( \varrho \) is punctual if it has a single time point, it is positive if it does not contain negative time points, and bounded if both of its endpoints are rationals numbers. For intervals \( \varrho \) and \( \varrho' \) we define the operations \( \varrho + \varrho' = \{ t + t' \mid t \in \varrho \text{ and } t' \in \varrho' \} \) and \( -\varrho = \{ -t \mid t \in \varrho \} \). We use the standard representation \( \langle \varrho^-, \varrho^+ \rangle \) for an interval \( \varrho \), where the left bracket \( \langle \) is either \( \{ \) or \( \{ , \text{ the right bracket } \) \( \rangle \) is either \( \) \( \text{ or } \) \( \} \), and \( \varrho^- \) and \( \varrho^+ \) are representations of the left and right endpoints of \( \varrho \), respectively. Brackets \( [ \) and \( ] \) indicate that the corresponding endpoints are included in the interval, whereas \( ( \) and \( ) \) indicate that they are not included. We often abbreviate a punctual interval \([t, t]\) as \( t \). Since different intervals cannot have the same representation, we will often abuse notation and identify an interval representation with the interval it represents.

Syntax. Assume a function-free first-order signature. A relational atom is a first-order atom of the form \( P(s) \), with \( P \) a predicate and \( s \) a tuple of whose length matches the arity of \( P \). A metric atom is an expression given by the following grammar, where \( P(s) \) is a relational atom and \( \varrho \) a positive interval:

\[
M ::= T | \bot | P(s) | \varrho M | \Phi_\varrho M | \\
\varrho M | \varrho M | MS_\varrho M | MU_\varrho M.
\]

A rule is an expression of the form

\[
M' \leftarrow M_1 \land \cdots \land M_n, \quad \text{for } n \geq 1, \quad (1)
\]

where each \( M_i \) is a metric atom, whereas \( M' \) is a metric atom not mentioning \( \varrho, \Phi, S, \) and \( U \), and hence generated by the following grammar:

\[
M' ::= T | P(s) | \varrho M' | \varrho M'.
\]

The conjunction \( M_1 \land \cdots \land M_n \) in Expression (1) is the rule’s body; each \( M_i \) is a body atom and \( M' \) is the rule’s head. A rule is safe if all its variables occur in the body.

A program is a finite set of safe rules. A program is bounded if all intervals it mentions are bounded. An expression (metric atom, rule, or program) is ground if it has no variables. The grounding ground(\( \Pi \)) of \( \Pi \) is the set of all ground rules obtained by assigning constants to variables in \( \Pi \). A metric fact over an interval \( \varrho \) is an expression \( M \circ \varrho \), with \( M \) a ground metric atom; it is relational if so is \( M \) and bounded if so is \( \varrho \). A dataset is a finite set of relational facts; it is bounded if so is each of its facts. For a dataset \( D \) we let \( t_D^- \) and \( t_D^+ \) be, respectively, the minimal and maximal rational numbers mentioned as endpoints of intervals in \( D \) (or 0 if \( D \) mentions no numbers).

Semantics. An interpretation \( \mathcal{I} \) specifies, for each ground relational atom \( P(s) \) and each time point \( t \), whether \( P(s) \) is satisfied at \( t \), in which case we write \( \mathcal{I}, t \models P(s) \). This notion extends to other ground metric atoms as specified in Table 1. Interpretation \( \mathcal{I} \) satisfies a metric fact \( M \circ \varrho \), written \( \mathcal{I} \models M \circ \varrho \), if \( \mathcal{I}, t \models M \) for all \( t \in \varrho \). Interpretation \( \mathcal{I} \) satisfies a ground rule \( r \) if, for any time point \( t \), whenever \( \mathcal{I} \) satisfies each body atom of \( r \) at \( t \), then \( \mathcal{I} \) also satisfies the head of \( r \) at \( t \). Interpretation \( \mathcal{I} \) satisfies a (possibly non-ground) rule \( r \) if \( \mathcal{I} \) satisfies each rule in ground(\( \{ r \} \)); it satisfies a program \( \Pi \) if \( \mathcal{I} \) satisfies each rule in \( \Pi \); moreover, \( \mathcal{I} \) satisfies \( \Pi \) in an interval \( \varrho \) if, for each \( r \in \text{ground}(\Pi) \) and each \( t \in \varrho \), whenever \( \mathcal{I} \) satisfies each body atom of \( r \) at \( t \), then \( \mathcal{I} \) satisfies the head of \( r \) at \( t \). Interpretation \( \mathcal{I} \) is a model of a program \( \Pi \) if it satisfies \( \Pi \), and it is a model of a dataset \( D \) if it satisfies each fact in \( D \). A program \( \Pi \) and a dataset \( D \) entail a metric fact \( M \circ \varrho \), written as \( (\Pi, D) \models M \circ \varrho \), if each model of both \( \Pi \) and \( D \) is also a model of \( M \circ \varrho \). We may write \( D \models M \circ \varrho \) instead of \( (\emptyset, D) \models M \circ \varrho \).

\(^2\)For convenience, we additionally disallow \( \bot \) in rule heads, which ensures consistency and allows us to focus on fact entailment and the computation of canonical interpretations.
An interpretation $\mathcal{I}$ contains an interpretation $\mathcal{I}'$, written $\mathcal{I}' \subseteq \mathcal{I}$, if $\mathcal{I}'(s) \models P(s)$ implies $\mathcal{I}(s) \models P(s)$, for each ground relational atom $P(s)$ and time point $t$. Whenever $\mathcal{I}$ satisfies each body atom of $r$ at a time point $t$, then $T^I_{\Pi}(\mathcal{I})$ satisfies the head of $r$ at $t$. Successive applications of $T^I_{\Pi}$ to $\mathcal{I}$ define a transfinite sequence of interpretations $T^\omega_{\Pi}(\mathcal{I})$ for ordinals $\alpha$ as follows: (i) $T^0_{\Pi}(\mathcal{I}) = \mathcal{I}$, (ii) $T^{\alpha+1}_{\Pi}(\mathcal{I}) = T^\alpha_{\Pi}(T^\alpha_{\Pi}(\mathcal{I}))$ for $\alpha$ an ordinal, and (iii) $T^n_{\Pi}(\mathcal{I}) = \bigcup_{\beta<n} T^\beta_{\Pi}(T^\beta_{\Pi}(\mathcal{I}))$ for $\alpha$ a limit ordinal. The canonical model $\mathcal{C}_{\Pi,\mathcal{D}}$ of $\Pi$ and $\mathcal{D}$ is the interpretation $T^\omega_{\Pi}(\mathcal{I})$, where $\omega$ is the first uncountable ordinal. Since we do not allow $\bot$ in rule heads, $\mathcal{C}_{\Pi,\mathcal{D}}$ is always a model of $\Pi$ and $\mathcal{D}$, and it is the least such model (Brandt et al. 2017). Interpretation $\mathcal{C}_{\Pi,\mathcal{D}}$ can be divided into regularly distributed $(\Pi, \mathcal{D})$-intervals whose time points satisfy the same ground atoms (Walega et al. 2019). In particular, the $(\Pi, \mathcal{D})$-ruler is the set of time points of the form $t + i \cdot \text{div}(\Pi)$, for $t \in \mathbb{Q}$ mentioned in $\mathcal{D}$ and $i \in \mathbb{Z}$, and where $\text{div}(\Pi) = \frac{1}{k}$, with $k$ being the product of the all denominators in the rational endpoints of the intervals mentioned in $\Pi$ (for definiteness, if $\Pi$ has no intervals with rational endpoints, we let $k = 1$ and hence $\text{div}(\Pi) = 1$).

A $(\Pi, \mathcal{D})$-interval is either a punctual interval over a time point on the $(\Pi, \mathcal{D})$-ruler or an interval $(t_1, t_2)$ with $t_1$ and $t_2$ consecutive time points on the $(\Pi, \mathcal{D})$-ruler. We say than an interpretation $\mathcal{I}$ is a $(\Pi, \mathcal{D})$-interpretation if, for every relational fact $M @ t$, it holds that $\mathcal{I} @ M$ implies $\mathcal{I} @ M @ \varrho$, where $\varrho$ is the $(\Pi, \mathcal{D})$-interval containing $t$. It turns out that $\mathcal{C}_{\Pi,\mathcal{D}}$ as well as $T^\omega_{\Pi}(\mathcal{I})$, for any ordinal $\alpha$, is a $(\Pi, \mathcal{D})$-interpretation (Walega et al. 2019).

**Reasoning.** We consider fact entailment, that is, checking if a relational fact is entailed by a program and a dataset. This problem is PSpace-complete in data complexity, that is, when the size of a program is considered as fixed (Wałęga et al. 2019), and ExpSpace-complete in combined complexity, where complexity is measured also with respect to the program (Brandt et al. 2017). The same complexity bounds hold already in the case of bounded programs and datasets (Wałęga, Zawidzki, and Cuenca Grau 2021).

**Periodic Structure of the Canonical Model**

In this section, we establish the theoretical basis for our novel reasoning approach. Given a bounded program $\Pi$ and a dataset $\mathcal{D}$, we present saturation conditions stating the properties that a partial materialisation $T^k_{\Pi}(\mathcal{D})$ needs to satisfy so that the full canonical model can be recovered. We call an interpretation satisfying these conditions saturated and show how to compute relevant periods and exploit them to unfold the interpretation into the canonical model of $\Pi$ and $\mathcal{D}$.

This allows us to check entailment of any fact.

For convenience of presentation, we fix for the remainder of this section an arbitrary bounded program $\Pi$, bounded dataset $\mathcal{D}$, and natural number $k$. Moreover, for an interpretation $\mathcal{I}$ and an interval $\varrho$, we let the projection $\mathcal{I} @ \varrho$ of $\mathcal{I}$ over $\varrho$ be the interpretation that coincides with $\mathcal{I}$ on $\varrho$ and makes all relational atoms false outside $\varrho$. We say that an interpretation $\mathcal{I}$ is a shift of $\mathcal{I}$ if there is a rational number $\varrho$ such that $\mathcal{I} = M @ \varrho$ if and only if $\mathcal{I}' = M @ (\varrho + q)$, for each (relational) fact $M @ \varrho$. Furthermore, for a rule $r$, we define its depth, written as $\text{depth}(r)$, as the sum of right endpoints of all intervals occurring in the operators of $r$ (or 0 if $r$ mentions no intervals), and we let $\text{depth}(\Pi)$ be the maximum depth of its rules. As shown in the following proposition, the depth of a bounded rule determines the time points that can be ‘affected’ by an application of this rule.

**Proposition 1.** For every interpretation $\mathcal{I}$, time point $t$, and bounded rule $r$, it holds that $\mathcal{I}$ satisfies $r$ at $t$ if and only if so does $\mathcal{I} @ [t - \text{depth}(r), t + \text{depth}(r)]$.

**Proof.** By definition, $\mathcal{I}$ satisfies $r$ at $t$ if and only if the body of $r$ does not hold at $t$ or the head of $r$ holds at $t$. By the definition of $\text{depth}(r)$, all the facts corresponding to the satisfaction of the body and the head of $r$ at $t$ are over intervals contained in $[t - \text{depth}(r), t + \text{depth}(r)]$. This directly implies the claim in the proposition.

Having introduced these basic concepts, we are now ready to define our notion of a saturated interpretation.

**Definition 2.** Interpretation $T^k_{\Pi}(\mathcal{D})$ is saturated if there exist closed intervals $\varrho_1, \varrho_2, \varrho_3$, and $\varrho_4$ of length $2 \text{depth}(\Pi)$, whose endpoints are located on the $(\Pi, \mathcal{D})$-ruler and satisfy $\varrho_1 < \varrho_2 < \varrho_3 < \varrho_4$, and such that the following properties hold:
- $T^k_{\Pi}(\mathcal{D})$ satisfies $\Pi$ in $[\varrho_1, \varrho_2]$;
- $T^k_{\Pi}(\mathcal{D}) \models \varrho_1$ and $T^k_{\Pi}(\mathcal{D}) \models \varrho_4$ are shifts of $T^k_{\Pi}(\mathcal{D}) \models \varrho_1$ and $T^k_{\Pi}(\mathcal{D}) \models \varrho_4$, respectively.

Any pair of intervals $[\varrho_1, \varrho_2]$ and $[\varrho_3, \varrho_4]$, for $\varrho_1, \varrho_2, \varrho_3, \varrho_4$ as above, will be referred to as periods of $T^k_{\Pi}(\mathcal{D})$. 
Intuitively, a saturated interpretation contains a ‘central fragment’ $[0^-, 0^+]$ which satisfies $D$ and such that a single application of $T_1$ does not derive any new facts within this fragment (i.e., the interpretation satisfies $II$ within $[0^-, 0^+]$). In the left segment of this ‘central fragment’ there are two intervals $0_1$ and $0_2$—each of length $2\text{depth}(II)$—in which $T_1^k(\Pi_D)$ satisfies the same relational facts modulo a shift. Analogously, in the right segment of the ‘central fragment’ there are intervals $0_3$ and $0_4$ also with repeating contents.

It is worth emphasising that in the definition of a saturated interpretation (first item), we consider only a single materialisation step, which can be effectively checked. As we will show in Theorem 5, rather surprisingly, this condition is enough to guarantee that a saturated interpretation can be unfolded into the canonical model.

We next define the unfolding of a saturated interpretation relatively to a pair $(\theta_{\text{left}}, \theta_{\text{right}})$ of its periods. Although there can be many such pairs of intervals, the key properties of the unfolding are independent of the choice of periods.

**Definition 3.** The $(\theta_{\text{left}}, \theta_{\text{right}})$-unfolding, of a saturated interpretation $T^k(\Pi_D)$ with periods $(\theta_{\text{left}}, \theta_{\text{right}})$, is the interpretation $C$ such that:

- $C |_{\theta_{\text{left}}^-} = T^k(\Pi_D) |_{\theta_{\text{left}}^-}$,
- $C |_{\theta_{\text{left}}^-} = T^k(\Pi_D) |_{\theta_{\text{left}}^-}$, for any $n \in \mathbb{N}$,
- $C |_{\theta_{\text{right}}^+} = T^k(\Pi_D) |_{\theta_{\text{right}}^+}$, for any $n \in \mathbb{N}$.

We observe that the unfolding is unique and can be obtained from the least interpretation coinciding with $T^k(\Pi_D)$ on $[\theta_{\text{left}}^-, \theta_{\text{right}}^+]$ by subsequently copying the fragment of this interpretation that spans $\theta_{\text{left}}$ infinitely many times to the left and the fragment of this interpretation that spans $\theta_{\text{right}}$ infinitely many times towards the right on the timeline.

In Theorem 5 we will show that the unfolding of a saturated interpretation coincides with the canonical model. The proof of this theorem exploits a generalisation of a technical lemma from the literature (Wałęga, Zawidzki, and Cuenca Grau 2021, Lemma 11), which we provide next.

**Lemma 4.** Let $\phi$ be a closed interval of length $\text{depth}(II)$ and $D'$ a dataset representing $\mathcal{C}_{II, D}$ $\models \phi$. Then both of the following hold:

1. If $\phi^+ \geq t^1_D$, then $\mathcal{C}_{II, D'} \models (\phi^+, \infty) \subseteq \mathcal{C}_{II, D'} \models (\phi^+, \infty)$.
2. If $\phi^- \leq t^1_D$, then $\mathcal{C}_{II, D'} \models (-\infty, \phi^-) \subseteq \mathcal{C}_{II, D'} \models (-\infty, \phi^-)$.

**Proof sketch.** Both items have similar proofs, so we focus on the first one. The inclusion $\mathcal{C}_{II, D'} \models (\phi^+, \infty) \subseteq \mathcal{C}_{II, D'} \models (\phi^+, \infty)$ follows from the fact that $\mathcal{C}_{II, D} \models D'$, so $\mathcal{C}_{II, D'} \models D'$. To show that $\mathcal{C}_{II, D'} \models (\phi^+, \infty) \subseteq \mathcal{C}_{II, D'} \models (\phi^+, \infty)$ it suffices to prove inductively that, for every ordinal $\alpha$ and relational fact $M \bo\theta t$ with $t > \phi^+$, if $T^k(\Pi_D) \models M \bo\theta t$, then $T^k(\Pi_D) \models M \bo\theta t$. We conduct the induction similarly to the proof due to Wałęga, Zawidzki, and Cuenca Grau (2021, Lemma 11), but using $\text{depth}(II)$ in the inductive step.

**Theorem 5.** The $(\theta_{\text{left}}, \theta_{\text{right}})$-unfolding $C$ of a saturated interpretation $T^k(\Pi_D)$ with periods $(\theta_{\text{left}}, \theta_{\text{right}})$ coincides with the canonical model $\mathcal{C}_{II, D}$.

**Proof sketch.** We first show that $\mathcal{C}_{II, D} \subseteq \mathcal{C}$. Since $\mathcal{C}_{II, D}$ is the least model of $II$ and $D$, it suffices to show that $\mathcal{C}$ is a model of $II$ and $D$. As $[t^1_D, t^2_D] \subseteq [\theta_{\text{left}}^+, \theta_{\text{right}}^+]$, we obtain that the projection of $T^k(\Pi_D)$ over $[\theta_{\text{left}}^+, \theta_{\text{right}}^+]$ is a model of $D$. However, by Definition 3, $\mathcal{C}$ and $T^k(\Pi_D)$ coincide on $[\theta_{\text{left}}^+, \theta_{\text{right}}^+]$, so $\mathcal{C}$ is also a model of $D$. To show that $\mathcal{C}$ satisfies $II$ in $\phi = [\theta_{\text{left}}^-, \theta_{\text{right}}^-]$ over $[\theta_{\text{left}}^-, \theta_{\text{right}}^-]$, it suffices, by Proposition 1, to show that the projection of $\mathcal{C}$ over $[\theta_{\text{left}}^-, \theta_{\text{right}}^-]$ satisfies $II$ in $\phi$, this holds since $\mathcal{C}$ and $T^k(\Pi_D)$ coincide on $[\theta_{\text{left}}^-, \theta_{\text{right}}^-]$ and $T^k(\Pi_D)$ satisfies $II$ in $[\theta_{\text{left}}^-, \theta_{\text{right}}^-]$, as it is saturated. It remains to argue that $\mathcal{C}$ satisfies $II$ at each $t$ outside $\phi$. Assume that $t < \theta^-$, so $t < \theta_{\text{left}} - \text{depth}(II)$. Then, there exists a unique pair of $n \in \mathbb{N}$ and $t' \in [\theta_{\text{left}} + \text{depth}(II), \theta_{\text{right}} + \text{depth}(II)]$ such that $t + n \cdot |\theta_{\text{left}}| = t'$. This, by the definition of $\mathcal{C}$, implies that $\mathcal{C} \models \phi$ is a shift of $\mathcal{C} \models \phi$ by $\text{depth}(II)$.

We show that $\mathcal{C} \subseteq \mathcal{C}_{II, D}$. The projection of $C$ over $[\theta_{\text{left}}^-, \theta_{\text{right}}^-]$ is in $\mathcal{C}_{II, D}$ since $C$ and $T^k(\Pi_D)$ coincide on this interval and $T^k(\Pi_D) \subseteq \mathcal{C}_{II, D}$. To show that the projection of $\mathcal{C}$ over $(-\infty, \theta_{\text{left}}^-)$ is in $\mathcal{C}_{II, D}$, we argue that $\mathcal{C} \models |_{[\theta_{\text{left}}^-, \theta_{\text{left}}^+]} \subseteq \mathcal{C}_{II, D} \models |_{[\theta_{\text{left}}^-, \theta_{\text{left}}^+]}$ by induction on consecutive timepoints $t_i$ on the $(II, D)$-ruler, with $t_0 = \theta_{\text{left}}^-$. The base holds since $C$ and $T^k(\Pi_D)$ coincide on $[\theta_{\text{left}}^-, \theta_{\text{right}}^+]$. For the inductive step, let $\theta_A = [t_i, t_{i-1}]$, $\theta_B = [t_{i-1}, t_{i-1} + \text{depth}(II)]$, $\theta_A' = \theta_A + |\theta_{\text{left}}^-|$, and $\theta_B' = \theta_B + |\theta_{\text{left}}^-|$. Hence, it suffices to show that $\mathcal{C} \models \phi_{\theta_A} \subseteq \mathcal{C}_{II, D} \models \phi_{\theta_A'}$. We note that $\mathcal{C} \models \phi_{\theta_A'}$ is a shift of $\mathcal{C} \models \phi_{\theta_A}$ by the construction of $\mathcal{C}$ and the fact that $\mathcal{C} \models \phi_{\theta_B} \subseteq \mathcal{C}_{II, D} \models \phi_{\theta_B'}$. By the inductive assumption, $\mathcal{C} \models \phi_{\theta_A} \subseteq \mathcal{C}_{II, D} \models \phi_{\theta_A'}$ and so, $\mathcal{C} \models \phi_{\theta_{\text{left}}^-} \subseteq \mathcal{C}_{II, D} \models \phi_{\theta_{\text{left}}^-}$ is symmetric; thus, $\mathcal{C} \subseteq \mathcal{C}_{II, D}$.}

Theorem 5 states that the unfolding of a saturated interpretation around any pair of its periods coincides with the canonical model. Thus, we can refer to an unfolding of a saturated interpretation without explicitly referring to its periods.

We conclude this section by establishing a bound $k_{\text{max}}$ depending on $II$ and $D$, which ensures that $T^k(\Pi_D)$ is saturated for some $k \leq k_{\text{max}}$. Intuitively, $k_{\text{max}}$ is the maximum of facts that we can construct by combining atoms relevant to $II$ and $D$ with $II$ and $D$-intervals contained in a sufficiently large interval $\phi$. The interval $\phi$ is chosen so that $[t^1_D, t^2_D] \subseteq \phi$, and such that its fragments $[\phi^-, t^1_D)$ and $(t^2_D, \phi^+]$ to the left...
and right of $D$, respectively, contain so many intervals of length $2\text{depth}(I)$ with endpoints on the $(I, D)$-ruler, that their contents need to repeat. These repetitions guarantee the existence of intervals $g_1, \ldots, g_4$ from Definition 2.

**Theorem 6.** There exists $k \leq k_{\text{max}}$ such that $T^k_{\Pi}(\mathcal{P}_D)$ is saturated, where the bound $k_{\text{max}}$ is defined as follows. Let $A$ be the number of ground relational atoms in the grounding of $\Pi$ with constants from $\Pi$ and $D$, let $B$ be the number of $(I, D)$-intervals within $[t_\ell, t_\ell + 2\text{depth}(I)]$, and let $q = [t_w - 2\text{depth}(I), t_w + 2\text{depth}(I)]$, where $\cdots < t_{-2} < t_{-1} < t_0$ and $t_0 \cdots < t_1 < t_2 < \ldots$ are sequences of consecutive time points on the $(I, D)$-ruler, with $w = 1 + B \cdot (2^A)^B$. Then $k_{\text{max}}$ is the product of $A$ and the number of $(I, D)$-intervals contained in $q$.

**Proof sketch.** The canonical model $\mathcal{E}_{\Pi, D}$ satisfies at most $k_{\text{max}}$ relational facts over $(I, D)$-intervals contained in $q$. Since each application of $T^k_{\Pi}$ (before reaching a fixpoint) introduces at least one new relational fact over an $(I, D)$-interval, there needs to exist $k \leq k_{\text{max}}$ such that $T^k_{\Pi}(\mathcal{P}_D) \models \tau^k_{\Pi}(\mathcal{P}_D)$. We argue that $T^k_{\Pi}(\mathcal{P}_D)$ is saturated for such $k$ by showing that $q$ contains intervals $g_1, \ldots, g_4$ satisfying the conditions from Definition 2. To show the existence of $g_1$ and $g_2$, we observe that $q$ contains $w = 1 + B \cdot (2^A)^B$ intervals of length $2\text{depth}(I)$ whose endpoints are located on the $(I, D)$-ruler to the left of $t_\ell$. We can show that there must exist amongst them two distinct intervals $g_1$ and $g_2$ such that $T^k_{\Pi}(\mathcal{P}_D) \models \tau^k_{\Pi}(\mathcal{P}_D) |_{g_1}$ is a shift of $T^k_{\Pi}(\mathcal{P}_D) \models \tau^k_{\Pi}(\mathcal{P}_D) |_{g_2}$. Indeed, each interval of the form $[t, t + 2\text{depth}(I)]$, for $t$ on the $(I, D)$-ruler, contains the same number of $(I, D)$-intervals as $[t, t + 2\text{depth}(I)]$ does, which equals $B$. In each of these $(I, D)$-intervals there can hold at most $2A$ combinations of relational atoms, which gives rise to $(2^A)^B$ different contexts of an interval of the form $[t, t + 2\text{depth}(I)]$. Additionally, these intervals can differ depending on the location of the $(I, D)$-intervals they contain. By the definition of the $(I, D)$-ruler, there are at most $B \cdot (2^A)^B$ intervals of the form $[t, t + 2\text{depth}(I)]$ with different contexts. Thus, in a set of $w = 1 + B \cdot (2^A)^B$ such intervals, there needs to be a pair of distinct intervals $g_1$ and $g_2$ such that $T^k_{\Pi}(\mathcal{P}_D) \models \tau^k_{\Pi}(\mathcal{P}_D) |_{g_1}$ is a shift of $T^k_{\Pi}(\mathcal{P}_D) \models \tau^k_{\Pi}(\mathcal{P}_D) |_{g_2}$. Analogously, we can show the existence of required $g_3$ and $g_4$ to the right of $t_\ell$, so the second item from Definition 2 holds. Moreover, since $[g_1^\ell, g_2^\ell] \subseteq \emptyset$ and $T^k_{\Pi}(\mathcal{P}_D) \models \tau^k_{\Pi}(\mathcal{P}_D) |_{g_3}$, we obtain that $T^k_{\Pi}(\mathcal{P}_D)$ satisfies $\Pi$ in $[g_1^\ell, g_2^\ell]$. Thus, the first item from Definition 2 holds as well, and so, $T^k_{\Pi}(\mathcal{P}_D)$ is saturated.

The bound $k_{\text{max}}$ from Theorem 6 is doubly exponential in the size of $\Pi$ and $D$, but only exponential in the size of $D$. Essentially, this bound shows us how much time is needed to perform reasoning, which is consistent with the computational complexity of reasoning in DataLogMTL, namely ExpSpace in combined complexity (Brandt et al. 2017) and PSpace in data complexity (Wałęga et al. 2019).

---

**Algorithm 1: Reasoning via Periods Detection**

**Input:** a bounded program $\Pi$, a bounded dataset $D$, and a relational fact $M \oplus q$

**Output:** a bounded program $\Pi$, a bounded dataset $D$, and a relational fact $M \oplus q$

1. $D_{\text{now}} := D$
2. loop
   1. if $D_{\text{now}} = M \oplus q$ then return True;
   2. $D_{\text{now}} := \text{ApplyRules}(\Pi, D_{\text{now}}, @\ell, @\ell, M \oplus q)$
   3. else return False;
3. if $D_{\text{now}} = M \oplus q$ then return True;
4. else return False;

**Algorithm**

The results from the previous section suggest the reasoning procedure in Algorithm 1, which checks if a bounded program $\Pi$ and a dataset $D$ entail a relational fact $M \oplus q$.

The algorithm successively applies the rules of $\Pi$ to the dataset (Line 8) until one of two stopping conditions hold. The first condition (Line 3) detects if the materialisation $D_{\text{now}}$ constructed thus far entails the input fact $M \oplus q$, in which case the algorithm reports that the input fact is entailed. The second condition checks if $D_{\text{now}}$ is saturated (i.e., represents a saturated interpretation). To this end, the algorithm computes in Line 4 two non-empty intervals which are periods of $D_{\text{now}}$ (if $D_{\text{now}}$ is saturated), or introduces two empty intervals (if $D_{\text{now}}$ is not saturated). If $D_{\text{now}}$ is saturated, the algorithm exploits $D_{\text{now}}$ and its periods to check whether $M \oplus q$ is entailed (Lines 6–7).

We next describe in detail the computations of the algorithm and establish their correctness. In what follows, we fix an arbitrary input $\Pi$, $D$, and $M \oplus q$ to Algorithm 1 and assume that the notions of a saturated interpretation and its periods are relative to $\Pi$ and $D$. We also let the projection $D' |q'$ of a dataset $D'$ over an interval $q'$ be the dataset obtained by intersecting all intervals in facts from $D'$ with $q'$.

**Rule Application** In each iteration of the loop, Algorithm 1 applies the procedure $\text{ApplyRules}$ (Line 8), which implements a single round of rule applications to $D_{\text{now}}$, and thus, mimics an application of the immediate consequence operator $T^k_{\Pi}$ to $D_{\text{now}}$. Hence, in the beginning of a $(k + 1)$st iteration of the loop, the dataset $D_{\text{now}}$ in Algorithm 1 represents $T^k_{\Pi}(\mathcal{P}_D)$. This is a basic component of materialisation in DataLogMTL, already implemented in the literature (Wałęga et al. 2022, Algorithm 1).

**First Stopping Condition** In Line 3, Algorithm 1 checks whether the materialisation $D_{\text{now}}$ constructed so far entails the input fact $M \oplus q$. Since facts in $D_{\text{now}}$ are stored in a coalesced form, to check if $D_{\text{now}} = M \oplus q$, it suffices to scan $D_{\text{now}}$ and verify whether there is $M \oplus q' \in D_{\text{now}}$ with $q \subseteq q'$.

**Second Stopping Condition** Algorithm 1 calls the $\text{Periods}$ procedure in Line 4 to check, in an iteration $k + 1$ of the loop, if $T^k_{\Pi}(\mathcal{P}_D)$ is saturated. The procedure searches
for intervals $\varrho_1, \ldots, \varrho_4$ satisfying the conditions in Definition 2 and returns the periods of the interpretation $T^k_{\Pi}(3_D)$ if it is saturated or a pair of empty intervals otherwise. To this end, the procedure performs one round of rule applications to $D_{\text{now}}$, computing $D_{\text{next}}$. Then, it computes an interval $\varrho_{\text{max}}$ as either the empty interval (if $D_{\text{now}}$ and $D_{\text{next}}$ do not coincide on $[t_D^-, t_D^+]$), or otherwise as the maximal interval containing $[t_D^-, t_D^+]$ and such that $D_{\text{now}}$ and $D_{\text{next}}$ coincide on $\varrho_{\text{max}}$. Interval $\varrho_{\text{max}}$ is next used to search for $\varrho_1$ and $\varrho_2$, namely all pairs of intervals contained in $\varrho_{\text{max}}$, located to the left of $t_D^-$, with endpoints on the $(\Pi, D)$-ruler, and of lengths $2^\text{depth}(\Pi)$, are compared. The first pair of such intervals with the same contents in $D_{\text{now}}$ is set as $\varrho_1$ and $\varrho_2$; otherwise $\varrho_1$ and $\varrho_2$ are empty intervals. In a similar way intervals $\varrho_3$ and $\varrho_4$ are computed. Finally, the procedure outputs a pair of intervals $(\varrho_1, \varrho_2), (\varrho_3, \varrho_4)$, or $(\emptyset, \emptyset)$ if any of the intervals $\varrho_1, \ldots, \varrho_4$ is empty.

**Fact Entailment Checking after Saturation**

After constructing a saturated dataset $D_{\text{now}}$ (representing $T^k_{\Pi}(3_D)$, for some $k \in \mathbb{N}$) with periods $(\varrho_{\text{left}}, \varrho_{\text{right}})$, Algorithm 1 calls the procedure Entails (Line 6) to check whether the input fact $M @ Q$ holds in the unfolding of $T^k_{\Pi}(3_D)$. This can be easily checked by exploiting Definition 3 of unfolding.

From our theoretical results in the previous section it follows that the algorithm is sound and complete:

**Theorem 7.** Algorithm 1 outputs True if $(\Pi, D) \models M @ Q$, otherwise it outputs False. Moreover, the algorithm terminates after at most $k_{\text{max}} + 1$ (c.f. Theorem 6) iterations of its main loop.

We conclude this section with an example illustrating the execution of Algorithm 1.

**Example 8.** Consider $\Pi = \{ \exists [0, 1] P \leftarrow P, \exists [1, 1] Q \leftarrow Q \}$, $D = \{ P @ 0, Q @ 1.5 \}$, and a query fact $M @ t = Q @ -4.5$. After 5 iterations of the loop in Algorithm 1, the dataset $D_{\text{now}}$ consists of facts $P @ 0, Q @ 1.5$ and $Q @ t$, for all $t \in \{-3.5, -2.5, -1.5, -0.5, 0.5, 1.5\}$. The stopping condition from Line 3 does not hold, but the condition in Line 5 does. Indeed, $D_{\text{now}}$ is saturated and its periods computed in Line 4 are $\varrho_{\text{left}} = [-3.5, -2.5)$ and $\varrho_{\text{right}} = (4.0, 4.5]$. The unfolding of $D_{\text{now}}$, $\varrho_{\text{left}}$, and $\varrho_{\text{right}}$ determine entailment of any fact. In particular, the input fact $Q @ -4.5$ is entailed, and so, Algorithm 1 returns True in Line 6.

**Experimental Evaluation**

We have implemented Algorithm 1 and conducted two experiments.\(^3\) The first experiment compares our implementation with MeTeoR (Wang et al. 2022), which we take as the baseline; we did not consider other reasoners, such as Ontop (Kalayci et al. 2018), as they support only non-recursive programs, where canonical interpretations are always finite. The second experiment tests the scalability of our implementation on datasets of increasing size. All experiments were conducted on a Dell PowerEdge R730 server with 512 GB of RAM and two Intel Xeon E5-2640 2.6 GHz processors running Fedora 33, kernel version 5.8.17. We used as benchmarks a temporal extension of the Lehigh University Benchmark (LUBM) (Wang et al. 2022) and the iTemporal benchmark generator (Bellomarini, Nissl, and Sallinger 2022). The LUBM extension provides a fixed DatalogMTL program which extends the 56 Datalog rules in LUBM with 29 temporal rules involving recursion and mentioning all metric operators in DatalogMTL. We used the iTemporal benchmark generator to construct another recursive DatalogMTL program with 19 rules. Both of the benchmarks allow for the generation of datasets of increasing size.

**Comparison with the Baseline.** We compared our implementation with MeTeoR, which combines materialisation with automata-based reasoning techniques. For the LUBM benchmark, Figure 2 (left), we considered a dataset $D$ with 5 million facts and query facts $F @ t$, where $F$ is a fixed relational atom (about the predicate FullProfessor) and $t \in \{-500, -400, \ldots, 500\}$. Reasoning with $D$ and the fragment $\Pi$ of the benchmark’s program that is relevant for $F$ is non-trivial and materialisation does not terminate for them.

Facts $F @ t$ with $t > 0$ are entailed by $\Pi$ and $D$, so such facts are decided by MeTeoR using materialisation only. Thus, the larger $t$, the larger the number of materialisation steps the baseline performs; as shown in Figure 2, materialisation times increase linearly with $t$. In our new approach, we observe a similar linear growth up until $t = 300$, where saturation is achieved. This indicates that saturation checks during materialisation come with negligible overhead. Once saturation is reached, our approach does not require further materialisation steps and fact entailment can be decided based on the saturated dataset. Thus, fact entailment running times are roughly identical for all facts with $t \geq 300$, while for MeTeoR they continue to grow with $t$.

Facts $F @ t$, with $t < 0$ are not entailed, hence MeTeoR must resort to automata-based techniques. Due to the optimisations implemented in MeTeoR, automata-based reasoning works well for facts with time points that lie close to the dataset, but performance degrades as $t$ becomes located further away. In contrast, our approach proceeds as in the previous case; it will materialise in linear running time until a saturated dataset is obtained, at which point running times stabilise and become independent of $t$.

In the case of the iTemporal benchmark, we considered a dataset with 1000 facts and facts $G @ t$ with a specific atom

---

\(^3\)Our implementation can be accessed through the following link: https://github.com/wdimmy/DatalogMTLPeriodicity.
Our approach
Increasing time range
benchmark (see Figure 3, right), which confirmed our ob-
our approach than increasing the temporal domain.

The number of constants seems to have a larger impact on
first dataset which is the same in both sequences), so increas-
around twice smaller than for the first sequence (except the
Running times for the second sequence of facts are
Scalability in Data Size.

In the second experiment, we
analysed how saturation times increase with the size of the
input dataset. We considered the same programs as in the
first experiment, together with sequences of datasets of in-
creasing size. For the LUBM benchmark, we generated two
sequences, each containing 6 datasets. In the first sequence
all the intervals of temporal facts are contained within the
range $[0, 50]$ and we increase the size of datasets by intro-
ducing atoms with new constants. In the second sequence,
the number of constants is the same but the number of facts
increases; facts occupy increasing ranges of time, namely,
$[0, 5 \cdot 10^i]$, for $i \in \{1, \ldots, 6\}$. In both sequences, the datasets
have $10^i$, $10^2$, $10^5$, $10^6$, $10^7$, and $10^8$ facts, respectively.

As depicted in Figure 3 (left), in both cases running times
grow proportionally with data size; this suggests that our
approach is scalable with respect to both the number of con-
stants and the temporal range of the input. We observe that
the running times for the second sequence of facts are
around twice smaller than for the first sequence (except the
first dataset which is the same in both sequences), so increas-
ing the number of constants seems to have a larger impact on
our approach than increasing the temporal domain.

We performed a similar experiment for the iTemporal
benchmark (see Figure 3, right), which confirmed our ob-
servations from the LUBM experiment.

Figure 2: Comparison with the baseline on the LUBM (on
the left) and iTemporal (on the right) benchmarks

Figure 3: Scalability of our approach tested on the LUBM
(on the left) and iTemporal (on the right) benchmarks

Scalability in Data Size. In the second experiment, we
analysed how saturation times increase with the size of the
input dataset. We considered the same programs as in the
first experiment, together with sequences of datasets of in-
creasing size. For the LUBM benchmark, we generated two
sequences, each containing 6 datasets. In the first sequence
all the intervals of temporal facts are contained within the
range $[0, 50]$ and we increase the size of datasets by intro-
ducing atoms with new constants. In the second sequence,
the number of constants is the same but the number of facts
increases; facts occupy increasing ranges of time, namely,
$[0, 5 \cdot 10^i]$, for $i \in \{1, \ldots, 6\}$. In both sequences, the datasets
have $10^i$, $10^2$, $10^5$, $10^6$, $10^7$, and $10^8$ facts, respectively.

As depicted in Figure 3 (left), in both cases running times
grow proportionally with data size; this suggests that our
approach is scalable with respect to both the number of con-
stants and the temporal range of the input. We observe that
the running times for the second sequence of facts are
around twice smaller than for the first sequence (except the
first dataset which is the same in both sequences), so increas-
ing the number of constants seems to have a larger impact on
our approach than increasing the temporal domain.

We performed a similar experiment for the iTemporal
benchmark (see Figure 3, right), which confirmed our ob-
servations from the LUBM experiment.

Figure 2: Comparison with the baseline on the LUBM (on
the left) and iTemporal (on the right) benchmarks

Figure 3: Scalability of our approach tested on the LUBM
(on the left) and iTemporal (on the right) benchmarks

Related Work

The complexity of reasoning in DatalogMTL and its frag-
ments has been studied in recent years (Brandt et al. 2018;
Walga et al. 2019, 2020b), as well as their alternative seman-
tics with favourable computational properties have been
proposed (Walga et al. 2020a; Ryzhikov, Walga, and Zakharyaschev 2019).
DatalogMTL has also been extended with stratified negation (Tena Cucala et al. 2021)
and with unrestricted negation under stable model semantics (Walga et al. 2021),
which is related to recent research on metric temporal ASP (Cabalar et al. 2020).

Several techniques have been proposed for reasoning
in DatalogMTL. Brzoska (1998) introduced proof systems
based on solving linear inequalities, MeTeoR (Wang et al.
2022) is based on an algorithm combining materialisation
and automata-based techniques, whereas Ontop implements
a reasoning algorithm based on query rewriting that is applicable
to non-recursive programs (Kalayc et al. 2018).

The use of blocking conditions to stop further application
of inference rules while retaining completeness is routinely
exploited in automated reasoning, especially in the context
of modal (Bolander and Blackburn 2007; Schmidt and Wald-
mann 2015; Areces and Orbe 2015), temporal (Reynolds
2016; Chaif et al. 2021), and description logics (Horrocks
and Sattler 1999; Schmidt and Tishkovsky 2011; Glimm,
Horrocks, and Motik 2010).

Periodicity of temporal models has been extensively in-
vestigated for LTL (Manna and Wolper 1982; Sistla and
Clarke 1985) and its fragments (Artale et al. 2013), the
temporal logic TPTL for the specification of real-time sys-
tems (Alur and Henzinger 1994), the temporalised descrip-
tion logic $\mathcal{EL}$ (Gutiérrez-Basulto, Jung, and Kontchakov
2016), and the metric temporal extension $\mathcal{LTL}_{\mathcal{A} \mathcal{C}}$ of the
description logic $\mathcal{A} \mathcal{C}$ (Baader et al. 2020). It has been shown
that canonical models in LTL-based rule-languages are peri-
odic and the lengths of offsets and periods in such interpreta-
tions are bounded by an exponential function of the number of
predicates involved (Artale et al. 2021b). In contrast to our
approach, however, no effective method of computing
such periods was established.

There have been various other proposals for extending
Datalog with temporal constructs (Baudinet, Chomicki, and
Wolper 1993; Beck et al. 2015) such as Datalog$^\text{TS}$ and
Datalog$^\text{nS}$ (Chomicki and Imieliński 1988; Chomicki
1990, 1995); in particular problems corresponding to finite mate-
rialisability have been considered in this setting (Chomicki
and Imieliński 1988).

Conclusions and Future Work

We have proposed a novel materialisation-based approach
for reasoning in DatalogMTL—a highly expressive
extension of Datalog with operators from metric temporal logic. Our
algorithm achieves termination by exploiting a specific
saturation condition, which ensures that materialisation
can be stopped after a bounded number of rounds of rule appli-
cations, without compromising completeness of reasoning.
We have implemented our approach and conducted experi-
ments supporting its feasibility in practice.
Acknowledgments

This research was supported in whole or in part by the EPSRC projects OASIS (EP/S032347/1), ConCuR (EP/V050869/1) and UK FIRES (EP/S019111/1), the SIRUS Centre for Scalable Data Access, and Samsung Research UK. For the purpose of Open Access, the authors have applied a CC BY public copyright licence to any Author Accepted Manuscript (AAM) version arising from this submission.

References


