Proceedings of the 18th International Conference on Principles of Knowledge Representation and Reasoning
Main Track

Finitely Materialisable Datalog Programs with Metric Temporal Operators

Przemysław A. Wałęga , **Michał Zawidzki** , **Bernardo Cuenca Grau**
Department of Computer Science, University of Oxford, UK
{przemyslaw.walega, michal.zawidzki, bernardo.cuenca.grau}@cs.ox.ac.uk

Abstract

DatalogMTL is an extension of Datalog with metric temporal operators that has recently received significant attention. In contrast to plain Datalog, where scalable implementations are often based on materialisation (a.k.a. forward chaining), reasoning algorithms for recursive fragments of DatalogMTL are automata-based and not well suited for practice. In this paper we propose the class of finitely materialisable DatalogMTL programs, for which forward chaining reasoning terminates after finitely many rounds of rule application. We show that, for bounded programs (a large fragment of DatalogMTL where temporal intervals are restricted to not mention infinity), checking whether a program is finitely materialisable is feasible in exponential time, and propose sufficient conditions for finite materialisability that can be checked more efficiently. We finally show that fact entailment over finitely materialisable bounded programs is ExpTime-complete, and hence no harder than Datalog reasoning.

1 Introduction

DatalogMTL is an extension of Datalog, where atoms in rules can include operators from metric temporal logic (MTL) usually interpreted over the rational timeline (Brandt et al. 2018; Wałęga et al. 2019). DatalogMTL is a powerful rule-based language for representing and reasoning about data and knowledge involving temporal intervals, with applications in stream reasoning (Wałęga, Cuenca Grau, and Kaminski 2019), temporal ontology-based query answering (Artale et al. 2017; Kikot et al. 2018), and knowledge graphs with temporal information (Vrandecic and Krötzsch 2014).

The study of the model-theoretic properties and computational complexity of DatalogMTL has recently received significant attention. Consistency checking and fact entailment are of high complexity, namely ExpSpace-complete in combined complexity (Brandt et al. 2018) and PSpace-complete in data (Wałęga et al. 2019) over both the rational and integer timelines (Wałęga et al. 2020a), although lower complexity fragments have also been identified (Wałęga et al. 2020b; Brandt et al. 2018; Wałęga et al. 2020a).

The aforementioned complexity upper bounds for reasoning have been mostly established using automata-based algorithms with comparable best-case and worst-case running times, and thus not well suited for efficient implementation. In contrast, scalable Datalog implementations often *materialise* (i.e., precompute using forward chaining via multiple rounds of rule application) all facts entailed by an input program and data. The facts in the materialisation provide a representation of the canonical, least model of the input over which all queries can be directly answered (Abiteboul, Hull, and Vianu 1995; Motik et al. 2019).

As in plain Datalog, in DatalogMTL each satisfiable pair of a program and dataset admit also a canonical model defined as the least fixpoint of an immediate consequence operator capturing a single round of rule application (Brandt et al. 2018; Wałęga et al. 2019). Consider, for instance, a program Π consisting of rule $\boxplus_{1year} Bday(x) \leftarrow Bday(x)$, which states that anyone having their birthday at a time point t will also be having their birthday at the same time the following year. Let now $\mathcal{D}$ be a dataset with a single fact saying that Alan Turing was having his first birthday during the 23rd of June 1913. In the canonical model of Π and $\mathcal{D}$, atom $Bday(\mathsf{Turing})$ holds at each time within June 23rd of each year from 1913 onwards; the first application of the rule makes $Bday(\mathsf{Turing})$ true during the 23rd of June 1914, and each subsequent application makes it true on the same day the year after. Hence, in contrast to Datalog, constructing a materialisation in DatalogMTL via forward chaining may require infinitely many rounds of rule application. Although a materialisation-based algorithm for non-recursive programs can be obtained from the results in (Brandt et al. 2018), to the best of our knowledge such algorithms for recursive fragments of DatalogMTL are yet to be developed.

In this paper we propose and study *finitely materialisable* DatalogMTL programs, for which forward chaining is guaranteed to construct a materialisation in a finite number of steps. Such programs are thus naturally well suited for materialisation-based reasoning, which paves the way to the development of efficient implementations. Finitely materialisable programs extend both Datalog, where programs may be recursive but do not contain metric operators (Abiteboul, Hull, and Vianu 1995), and non-recursive DatalogMTL, where the use of metric operators is unrestricted but there are no cyclic dependencies between predicates (Brandt et al. 2018). Indeed, finite materialisability can be seen as a safe form of *temporal recursion*, thus capturing a form of temporal *boundedness* (Cosmadakis et al. 1988).

The main decision problems we consider are

– *finite materialisability*, which is to check whether a pro-

gram is finitely materialisable; and

– *fact entailment*, which is to check whether a finitely materialisable program and a dataset entail a given fact.

We focus on the fragment of DatalogMTL where all intervals in programs and in datasets are bounded (i.e., $-\infty$ and ∞ are not mentioned). We prove that finite materialisability in this setting is decidable in ExpTime, by showing that the problem reduces to considering the application of forward chaining to a single *critical dataset*. Indeed, finite materialisability bears some resemblance to checking universal termination of *chase* procedures for extensions of Datalog with existential quantifiers or function symbols (Gogacz and Marcinkowski 2014; Cuenca Grau et al. 2013; Marnette and Geerts 2010; Grahne and Onet 2018; Krötzsch, Marx, and Rudolph 2019; Baget et al. 2014), as well as to formal tools for verifying strong or weak *safety* of temporal programs (Chomicki and Imieliński 1988; Chomicki 1990; Chomicki 1995), where techniques based on identifying critical datasets are routinely exploited. To address this high complexity, we also provide a sufficient condition verifiable in coNP in general, and in NL for forward propagating and backwards propagating programs (Wałęga et al. 2019). Our condition is based on the analysis of a type of dependency graphs for programs, which is conceptually similar to techniques used for checking temporal acyclicity of description logics (Gutiérrez-Basulto, Jung, and Kontchakov 2016a; Gutiérrez-Basulto, Jung, and Kontchakov 2016b). Finally, we show that fact entailment in our setting is ExpTime-complete, and thus easier than reasoning over arbitrary bounded programs (which is ExpSpace-complete), and also no harder than reasoning in plain Datalog.

2 Preliminaries

We recapitulate the definition of DatalogMTL. We focus on the rational timeline and the continuous semantics (Brandt et al. 2018; Wałęga et al. 2019), as opposed to the integer timeline (Wałęga et al. 2020a) or the alternative pointwise semantics (Ryzhikov, Wałęga, and Zakharyaschev 2019).

Time and Intervals. The *(rational) timeline* is the 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* ϱ is a non-empty subset of $\mathbb{Q}$ satisfying the following 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 ϱ^- and the least upper bound ϱ^+ of ϱ belong to $\mathbb{Q} \cup \{-\infty, \infty\}$. The bounds ϱ^- and ϱ^+ are called the *left* and *right endpoints* of ϱ, respectively, and $\varrho^+ - \varrho^-$ is the *length* of ϱ. Interval ϱ is *punctual* if it contains exactly one number, it is *positive* if it does not contain negative numbers, and it is *bounded* if both its left and right endpoints are rational numbers. We use the standard representation $\langle \varrho^-, \varrho^+ \rangle$ for interval ϱ, where the *left bracket* $\langle$ is either [or (, the *right bracket* $\rangle$ is either] or), and ϱ^- and ϱ^+ are representations of the left and right endpoints of ϱ, respectively. As usual, the brackets [and] indicate that the corresponding endpoints are included in the interval, whereas (and) indicate that they are not included. We will often abbreviate a punctual interval $[t, t]$ as t. Since two different intervals cannot have the same representation, we will often abuse notation and identify each 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 $P(\mathbf{s})$, with P an n-ary predicate and $\mathbf{s}$ an n-ary tuple of terms. A *metric atom* is an expression given by the following grammar, where $P(\mathbf{s})$ is a relational atom and ϱ a positive interval:

$$M ::= \top \mid \bot \mid P(\mathbf{s}) \mid \diamondminus_\varrho M \mid \boxminus_\varrho M \mid \boxminus_\varrho M \mid \\ \boxminus_\varrho M \mid \boxplus_\varrho M \mid M \mathcal{S}_\varrho M \mid M \mathcal{U}_\varrho M.$$

A *rule* is an expression of the form

$$M' \leftarrow M_1 \wedge \cdots \wedge M_n, \qquad \text{for } n \geq 1, \tag{1}$$

where each M_i is a metric atom, whereas M' is a metric atom not mentioning $\diamondminus$, $\boxminus$, $\mathcal{S}$, and $\mathcal{U}$, and hence generated by the following grammar:[1]

$$M' ::= \top \mid P(\mathbf{s}) \mid \boxminus_\varrho M' \mid \boxplus_\varrho M'.$$

The conjunction $M_1 \wedge \cdots \wedge M_n$ in Expression (1) is the rule's *body*, where each M_i is a *body atom*, and M' is the rule's *head*. A rule is *safe* if all variables occur in the body. A *program* is a finite set Π of safe rules; it is propositional if its predicates are propositions (i.e., have arity 0). Program Π is *bounded* if all intervals it mentions are bounded and $\top$ does not occur in rule bodies.[2] The *dependency graph* of Π is a directed graph with a vertex v_P for each predicate P in Π and an edge (v_Q, v_R) if there is a rule mentioning Q in its body and R in its head. Program Π is *recursive* if its dependency graph is cyclic. An expression (metric atom, rule, or program) is *ground* if it mentions no variables. The *grounding* $\mathsf{ground}(\Pi)$ of Π is the set of ground rules obtained by assigning constants in the signature to variables in Π. We let $\mathsf{ground}(\Pi, \mathcal{D})$ denote the grounding of Π w.r.t. only the constants in Π and $\mathcal{D}$. A *metric fact* over interval ϱ is an expression $M@\varrho$, with M a ground metric atom; it is relational if so is M and bounded if so is ϱ. A *dataset* is a finite set of relational facts; it is bounded if so is each of its facts.

Semantics. An *interpretation* $\mathfrak{I}$ specifies, for each ground relational atom $P(\mathbf{s})$ and each time point t, whether $P(\mathbf{s})$ is *satisfied* at t, in which case we write $\mathfrak{I}, t \models P(\mathbf{s})$. This notion extends to other ground metric atoms as given in Table 1. Interpretation $\mathfrak{I}$ satisfies a metric fact $M@\varrho$, written $\mathfrak{I} \models M@\varrho$, if $\mathfrak{I}, t \models M$ for all $t \in \varrho$. Interpretation $\mathfrak{I}$ satisfies a ground rule r if, whenever $\mathfrak{I}$ satisfies each body atom of r at a time point t, then $\mathfrak{I}$ also satisfies the head of r at t. Interpretation $\mathfrak{I}$ satisfies a rule r if it satisfies each rule in $\mathsf{ground}(\{r\})$. Interpretation $\mathfrak{I}$ is a *model* of a program Π if it satisfies each rule in Π, and it is a *model* of a set of metric facts if it satisfies each of these facts. A set $\mathcal{M}$ of metric

[1] For presentational convenience, we disallow $\bot$ in rule heads, which ensures satisfiability and allows us to focus on fact entailment and the computation of canonical models.

[2] A rule $P \leftarrow \top$ simulates a fact $P@(-\infty, \infty)$ over an unbounded interval.

$\mathfrak{I}, t \models \top$	for each t
$\mathfrak{I}, t \models \bot$	for no t
$\mathfrak{I}, t \models \Diamond_\varrho M$	iff $\mathfrak{I}, t' \models M$ for some t' with $t - t' \in \varrho$
$\mathfrak{I}, t \models \Diamond_\varrho M$	iff $\mathfrak{I}, t' \models M$ for some t' with $t' - t \in \varrho$
$\mathfrak{I}, t \models \boxminus_\varrho M$	iff $\mathfrak{I}, t' \models M$ for all t' with $t - t' \in \varrho$
$\mathfrak{I}, t \models \boxplus_\varrho M$	iff $\mathfrak{I}, t' \models M$ for all t' with $t' - t \in \varrho$
$\mathfrak{I}, t \models M_1 \mathcal{S}_\varrho M_2$	iff $\mathfrak{I}, t' \models M_2$ for some t' with $t - t' \in \varrho$ and $\mathfrak{I}, t'' \models M_1$ for all $t'' \in (t', t)$
$\mathfrak{I}, t \models M_1 \mathcal{U}_\varrho M_2$	iff $\mathfrak{I}, t' \models M_2$ for some t' with $t' - t \in \varrho$ and $\mathfrak{I}, t'' \models M_1$ for all $t'' \in (t, t')$

Table 1: Semantics of ground metric atoms

facts *entails* a metric fact $M@\varrho$ if each model of $\mathcal{M}$ is also a model of $M@\varrho$. A program Π and a set $\mathcal{M}$ of metric facts *entail* a set $\mathcal{M}'$ of metric facts, written $(\Pi, \mathcal{M}) \models \mathcal{M}'$, if each model of both Π and $\mathcal{M}$ is also a model of $\mathcal{M}'$. We may write $\mathcal{M} \models \mathcal{M}'$ instead of $(\emptyset, \mathcal{M}) \models \mathcal{M}'$. If $\mathcal{M}$ or $\mathcal{M}'$ is a singleton, say $\{M@\varrho\}$, then we may omit curly brackets and write $M@\varrho \models \mathcal{M}'$ and $\mathcal{M} \models M@\varrho$, respectively.

Interpretation $\mathfrak{I}$ *contains* interpretation $\mathfrak{I}'$, written $\mathfrak{I}' \subseteq \mathfrak{I}$, if $\mathfrak{I}', t \models P(\mathbf{s})$ implies $\mathfrak{I}, t \models P(\mathbf{s})$, for each ground relational atom $P(\mathbf{s})$ and time point t. Then, $\mathfrak{I}$ is the *least* interpretation in a set X of interpretations if $\mathfrak{I} \subseteq \mathfrak{I}'$ for every $\mathfrak{I}' \in X$. Each dataset $\mathcal{D}$ admits the least interpretation $\mathfrak{I}_\mathcal{D}$ among all models of $\mathcal{D}$ and we say that a dataset $\mathcal{D}$ *represents* an interpretation $\mathfrak{I}$ if $\mathfrak{I} = \mathfrak{I}_\mathcal{D}$.

Canonical Interpretation The *immediate consequence operator* T_Π for a program Π is a function mapping an interpretation $\mathfrak{I}$ to the least interpretation $T_\Pi(\mathfrak{I})$ containing $\mathfrak{I}$ and satisfying the following property for each $r \in \mathsf{ground}(\Pi)$: whenever $\mathfrak{I}$ satisfies each body atom of r at a time point t, then $T_\Pi(\mathfrak{I})$ satisfies the head of r at t. The successive application of T_Π to $\mathfrak{I}_\mathcal{D}$ defines a transfinite sequence of interpretations $T^\alpha_\Pi(\mathfrak{I}_\mathcal{D})$ for ordinals α as follows: (i) $T^0_\Pi(\mathfrak{I}_\mathcal{D}) = \mathfrak{I}_\mathcal{D}$, (ii) $T^{\alpha+1}_\Pi(\mathfrak{I}_\mathcal{D}) = T_\Pi(T^\alpha_\Pi(\mathfrak{I}_\mathcal{D}))$ for α an ordinal, and (iii) $T^\alpha_\Pi(\mathfrak{I}_\mathcal{D}) = \bigcup_{\beta<\alpha} T^\beta_\Pi(\mathfrak{I}_\mathcal{D})$ for α a limit ordinal. The *canonical interpretation* $\mathfrak{C}_{\Pi,\mathcal{D}}$ of Π and $\mathcal{D}$ is the interpretation $T^{\omega_1}_\Pi(\mathfrak{I}_\mathcal{D})$, where ω_1 is the first uncountable ordinal. Since we do not allow $\bot$ in rule heads, $\mathfrak{C}_{\Pi,\mathcal{D}}$ is the least model of Π and $\mathcal{D}$ (Brandt et al. 2017). Interpretation $\mathfrak{C}_{\Pi,\mathcal{D}}$ can be divided into regularly distributed $(\Pi, \mathcal{D})$-*intervals* whose time points satisfy the same ground atoms (Wałęga et al. 2019). In particular, the $(\Pi, \mathcal{D})$-*ruler* is the set of time points of the form $t + i \cdot \mathsf{div}(\Pi)$, for $t \in \mathbb{Q}$ mentioned in $\mathcal{D}$ and $i \in \mathbb{Z}$, and where $\mathsf{div}(\Pi) = \frac{1}{k}$, with k the product of all denominators in the rational endpoints of the intervals mentioned in Π; if Π has no intervals with rational endpoints, then $k = 1$ (and hence $\mathsf{div}(\Pi) = 1$) for definiteness. A $(\Pi, \mathcal{D})$-*interval* is either a punctual interval over a time point on the ruler, or an interval (t_1, t_2) with t_1 and t_2 consecutive time points on the ruler. For every Π, $\mathcal{D}$, and relational fact $M@t$, if $\mathfrak{C}_{\Pi,\mathcal{D}} \models M@t$, then $\mathfrak{C}_{\Pi,\mathcal{D}} \models M@\varrho$, with ϱ the $(\Pi, \mathcal{D})$-interval containing t (Wałęga et al. 2019).

Reasoning. We consider *fact entailment*: checking if a relational fact is entailed by a program and a dataset. For arbitrary DatalogMTL programs, fact entailment (in combined complexity) is ExpSpace-complete and for for non-recursive programs it is PSpace-complete (Brandt et al. 2017).

3 Finitely Materialisable Programs

In this section we introduce the notion of *finitely materialisable programs*. Our definition is rather general. On the one hand, it is based on the semantics of the immediate consequence operator and hence is not tied to any syntactic materialisation procedure computing a representation of the canonical model; on the other hand, it is parametrised by a class $\mathbf{C}$ of datasets, where natural instantiations include the classes of all datasets, all bounded datasets, or all punctual datasets.

Definition 1. *Let Π be a program and α an ordinal number. The immediate consequence operator T_Π* converges for a dataset $\mathcal{D}$ in α steps *if $T^\alpha_\Pi(\mathfrak{I}_\mathcal{D}) = \mathfrak{C}_{\Pi,\mathcal{D}}$. Moreover, Π is* finitely materialisable *for a class $\mathbf{C}$ of datasets if for each $\mathcal{D} \in \mathbf{C}$ operator T_Π converges in a finite number of steps.*

For instance, our example program Π from Section 1 consisting of rule $\boxplus_1 Bday(x) \leftarrow Bday(x)$ is not finitely materialisable (for any reasonable class of datasets). Indeed, T_Π does not converge in finitely many steps for any dataset $\mathcal{D}$ of the form $\{Bday(c)@\varrho\}$, with c an arbitrary constant and ϱ an arbitrary interval. This is because the rule of Π is recursive and involves a metric operator propagating facts to the future. As we show next, recursion is a necessary condition for a program to be not finitely materialisable.

Proposition 2. *Non-recursive* DatalogMTL *programs are finitely materialisable for every class of datasets.*

Proof. We show the contrapositive. Assume that Π is not finitely materialisable for some dataset class $\mathbf{C}$. Then, there is $\mathcal{D} \in \mathbf{C}$ such that $T^k_\Pi(\mathfrak{I}_\mathcal{D}) \neq T^{k-1}_\Pi(\mathfrak{I}_\mathcal{D})$ for each $k > 0$. Let $k > 0$; since $T^k_\Pi(\mathfrak{I}_\mathcal{D}) \neq T^{k-1}_\Pi(\mathfrak{I}_\mathcal{D})$, there is a relational fact $M_k@t_k$ holding in $T^k_\Pi(\mathfrak{I}_\mathcal{D})$ but not in $T^{k-1}_\Pi(\mathfrak{I}_\mathcal{D})$. We construct a sequence $M_1@t_1, \ldots, M_k@t_k$ of facts, where each M_i is a relational atom, $M_i@t_i$ holds in $T^i_\Pi(\mathfrak{I}_\mathcal{D})$ but not in $T^{i-1}_\Pi(\mathfrak{I}_\mathcal{D})$, and there is a rule in $\mathsf{ground}(\Pi)$ with M_{i-1} in the body and M_i in the head. Assume that $M_i@t_i$ is given, for $i \geq 2$, and we want to construct $M_{i-1}@t_{i-1}$. Since $M_i@t_i$ holds in $T^i_\Pi(\mathfrak{I}_\mathcal{D})$ but not in $T^{i-1}_\Pi(\mathfrak{I}_\mathcal{D})$, there is $r \in \mathsf{ground}(\Pi)$ and $t \in \mathbb{Q}$ such that all body atoms of r hold at t in $T^{i-1}_\Pi(\mathfrak{I}_\mathcal{D})$ and the head holding at t entails $M_i@t_i$. If the body of r did hold at t in $T^{i-2}_\Pi(\mathfrak{I}_\mathcal{D})$, then $M_i@t_i$ would hold in $T^{i-1}_\Pi(\mathfrak{I}_\mathcal{D})$, which raises a contradiction. Hence, there is a relational atom M in the body of r, which holds at some t' in $T^{i-1}_\Pi(\mathfrak{I}_\mathcal{D})$ but not in $T^{i-2}_\Pi(\mathfrak{I}_\mathcal{D})$ (since $i \geq 2$, we can show that $M \neq \top$). Thus, we let $M_{i-1}@t_{i-1}$ be $M@t'$.

The existence of this sequence of facts implies that the dependency graph of Π has a path of length at least $k-1$. Since k is arbitrary, the graph is cyclic, and so, Π is recursive. □

The presence of recursion via metric atoms is, however, not always harmful, and many recursive programs that are interesting in practice are finitely materialisable.

Example 3. *There is growing evidence that individuals vaccinated against COVID-19 who remain without symptoms 3 to 4 weeks following vaccination develop immunity. Furthermore, individuals with a negative test taken 3 to 4 weeks after vaccination, or who were infected within the last 6 months (discounting the last ten days when they had no symptoms) are also immune. Finally, those immune for the last 5 days display a negative test result. These conditions can be captured by a* DatalogMTL *program with the following rules:*

$$\mathit{Immune}(x) \leftarrow \mathit{NoSympt}(x)\mathcal{S}_{[21,28]}\mathit{Vaccinated}(x),$$
$$\mathit{Immune}(x) \leftarrow \mathit{NegTest}(x) \wedge \diamondsuit_{[21,28]}\mathit{Vaccinated}(x),$$
$$\mathit{Immune}(x) \leftarrow \diamondsuit_{(10,183]}\mathit{Infected}(x) \wedge \boxminus_{[0,10]}\mathit{NoSympt}(x),$$
$$\mathit{NegTest}(x) \leftarrow \boxminus_{[0,5]}\mathit{Immune}(x).$$

Although the program is recursive (its dependency graph is cyclic), it can be checked that the materialisation for any dataset can be reached after one round of rule application.

In what follows we show that finitely materialisable programs are amenable to forward chaining reasoning on any dataset within the relevant class, in the sense that forward chaining constructs a dataset representing their canonical model. The key observation is that, given a program Π and a dataset $\mathcal{D}$, we can always construct a dataset that represents the interpretation resulting from applying once T_Π to $\mathfrak{I}_\mathcal{D}$.

Theorem 4. *There is an algorithm that takes as an input a program Π and a dataset $\mathcal{D}$, and constructs a dataset $\mathcal{D}'$ representing $T_\Pi(\mathfrak{I}_\mathcal{D})$.*

Proof sketch. The algorithm performs the following steps:

1. Compute the set B of all ground body atoms (and their subformulas) occurring in rules of $\mathsf{ground}(\Pi, \mathcal{D})$.
2. Construct a finite set $\mathcal{B}$ of metric facts satisfying the following: $\mathcal{D} \models \mathcal{B}$, and if $\mathcal{D} \models M@t$ for some $M \in B$ and $t \in \mathbb{Q}$, then there exists ϱ such that $t \in \varrho$ and $M@\varrho \in \mathcal{B}$.
3. Construct $\mathcal{D}'$ by initially setting it to be $\mathcal{D}$ and then, for each $r \in \mathsf{ground}(\Pi, \mathcal{D})$ which mentions some relational atom in its head (otherwise r does not allow us do derive any new facts), by adding to $\mathcal{D}'$ facts as follows:
   - Use $\mathcal{B}$ to compute a finite set I_r of intervals for which all body atoms of r hold in $\mathfrak{I}_\mathcal{D}$.
   - For each $\varrho \in I_r$, compute the largest (with respect to set inclusion) ϱ' such that $H@\varrho \models P(\mathbf{s})@\varrho'$, where H is the head of r and $P(\mathbf{s})$ the unique relational atom in H. Add $P(\mathbf{s})@\varrho'$ to $\mathcal{D}'$.

The set B from Step 1 is constructed simply by scanning $\mathsf{ground}(\Pi, \mathcal{D})$. The set $\mathcal{B}$ from Step 2 is constructed inductively on the structure of metric atoms M in B. If M is $\top$, we add $\top@(-\infty, \infty)$ to $\mathcal{B}$, and if M is relational, then for each fact $M@\varrho$ in $\mathcal{D}$, we add $M@\varrho$ to $\mathcal{B}$. If $M = \diamondsuit_\varrho M'$, then for each fact $M'@\varrho' \in \mathcal{B}$, we compute the largest ϱ'' such that $M'@\varrho' \models \diamondsuit_\varrho M'@\varrho''$ and we add $\diamondsuit_\varrho M'@\varrho''$ to $\mathcal{B}$. If $M = \boxminus_\varrho M'$, we construct a set X of metric facts by coalescing facts of the form $M'@\varrho'$ in $\mathcal{B}$ (coalescing $M'@\varrho_1$ and $M'@\varrho_2$ yields $M'@\varrho_1 \cup \varrho_2$ if $\varrho_1 \cup \varrho_2$ is an interval); then, for each $M'@\varrho' \in X$, we compute the largest ϱ'' (if exists) such that $M'@\varrho' \models \boxminus_\varrho M'@\varrho''$ and add $\boxminus_\varrho M'@\varrho''$ to $\mathcal{B}$. If $M = M_1 \mathcal{S}_\varrho M_2$, then we construct—as in the previous case—a set X by coalescing facts of the form $M_1@\varrho_1$ in $\mathcal{B}$. For every pair of facts $M_1@\varrho_1 \in X_1$ and $M_2@\varrho_2 \in \mathcal{B}$, we compute the largest interval ϱ'' (if it exists) such that $\{M_1@\varrho_1, M_2@\varrho_2\} \models M_1 \mathcal{S}_\varrho M_2@\varrho''$ and we add $M_1 \mathcal{S}_\varrho M_2@\varrho''$ to $\mathcal{B}$. The construction for atoms with $\oplus$, $\boxplus$, and $\mathcal{U}$ is analogous to the construction for $\diamondsuit$, $\boxminus$, and $\mathcal{S}$, respectively. Note that $\mathcal{B}$ allows us to check in which intervals bodies of rules from $\mathsf{ground}(\Pi, \mathcal{D})$ hold. The construction of I_r from the Step 3 of the algorithm is obtained by scanning facts in $\mathcal{B}$, and given $\varrho \in I_r$, we determine ϱ' from the form of metric operators in the head of r.

The output $\mathcal{D}'$ extends $\mathcal{D}$ with a finite set of relational facts which are obtained by a single iteration of applying rules from Π to $\mathcal{D}$, and so, $\mathfrak{I}_{\mathcal{D}'} = T_\Pi(\mathfrak{I}_\mathcal{D})$. The procedure terminates: the constructions of $\mathcal{B}$ and $\mathcal{D}'$ have finitely many steps and in each step a finite number of facts are added. □

Algorithm 1: Materialisation-based reasoning

Input: A program Π which is finitely materialisable for a class $\mathbf{C}$ and a dataset $\mathcal{D} \in \mathbf{C}$
Output: A dataset $\mathcal{D}'$

```
Assign D' := D ;
loop
    Assign D_aux := D' ;
    Assign D' := the dataset representing T_Π(I_D')
      obtained by the algorithm from Theorem 4 ;
    if D_aux ⊨ D' then
        Return D'
```

Theorem 4 suggests a generic forward chaining reasoning procedure as presented in Algorithm 1. In particular, given a program Π which is finitely materialisable for a class $\mathbf{C}$ and a dataset $\mathcal{D} \in \mathbf{C}$ as an input, Algorithm 1 uses the procedure from the proof of Theorem 4 to iteratively compute a (finite) sequence of datasets $\mathcal{D}_0, \mathcal{D}_1, \ldots, \mathcal{D}_k$, where $\mathcal{D}_0 = \mathcal{D}$, each $\mathcal{D}_i$ represents the interpretation $T_\Pi(\mathfrak{I}_{\mathcal{D}_{i-1}})$, and $\mathcal{D}_k$ is the output dataset $\mathcal{D}'$ representing the canonical model $\mathfrak{C}_{\Pi,\mathcal{D}}$.

Corollary 5. *Algorithm 1 outputs a dataset representing the canonical model of its input.*

Note also that the termination of Algorithm 1 for finitely materialisable programs implies that the canonical model of such a program and any dataset can be finitely represented.

Corollary 6. *For every program Π finitely materialisable for a class $\mathbf{C}$ and for every $\mathcal{D} \in \mathbf{C}$ there exists a dataset $\mathcal{D}'$ which represents $\mathfrak{C}_{\Pi,\mathcal{D}}$.*

The converse, however, may not hold: programs admitting a finite representation of their canonical model in conjunction with any dataset may not be finitely materialisable.

Example 7. *Consider a program* $\Pi = \{P \leftarrow \diamondminus_{[0,1]} P\}$ *stating that, if* P *is true at some* t*, then it is true at the interval* $[t, t+1]$*. Consider also* $\mathcal{D} = \{P@0\}$*, stating that* P *is true at* 0*. The canonical model of* Π *and* $\mathcal{D}$ *makes* P *true at each* $t \geq 0$*; thus, it can be finitely represented by a single fact* $P@[0, \infty)$*. However,* T_Π *does not converge for* $\mathcal{D}$ *in any finite number of steps. Indeed, the first application of* T_Π *to* $\mathfrak{I}_\mathcal{D}$ *entails* $P@[0, 1]$*, the second* $P@[0, 2]$*, and so on.*

In the remainder of this paper, we study the main decision problems in our setting: finite materialisability with respect to a class of datasets, and fact entailment over finitely materialisable programs. We focus on a fragment of bounded DatalogMTL programs and the class of bounded datasets, thus ensuring that all intervals mentioned in programs and data have rational endpoints. This is a large fragment of DatalogMTL for which reasoning is as hard as for the full language—which can be shown by a slight modification of an existing hardness proof (Brandt et al. 2018).

Proposition 8. *Checking whether a bounded program and a bounded dataset entail a bounded fact is* ExpSpace*-hard.*

We leave for future work the investigation of these problems for unrestricted DatalogMTL programs and datasets, as well as for other interesting fragments of DatalogMTL.

4 Checking Finite Materialisability

In this section we study checking finite materialisability of bounded programs for the class of bounded datasets.

The restriction to bounded intervals in programs Π and datasets $\mathcal{D}$ provides a useful characterisation of finite convergence of T_Π for $\mathcal{D}$ in terms of the number of relational facts over $(\Pi, \mathcal{D})$-intervals entailed by Π and $\mathcal{D}$.

Lemma 9. *Let* Π *be a bounded program and* $\mathcal{D}$ *a bounded dataset. The operator* T_Π *converges for* $\mathcal{D}$ *in finitely many steps if and only if* Π *and* $\mathcal{D}$ *entail finitely many relational facts over* $(\Pi, \mathcal{D})$*-intervals.*

Proof sketch. Assume that $T_\Pi^k(\mathfrak{I}_\mathcal{D}) = \mathfrak{C}_{\Pi,\mathcal{D}}$ for some finite k. Since Π is bounded, we can show that interpretations $T_\Pi^i(\mathfrak{I}_\mathcal{D})$ entail finitely many relational facts over $(\Pi, \mathcal{D})$-intervals, for each $i \in \{0, \dots, k\}$. In particular, $\mathcal{D}$ is bounded, so $T_\Pi^0(\mathfrak{I}_\mathcal{D}) = \mathfrak{I}_\mathcal{D}$ entails finitely many relational facts over $(\Pi, \mathcal{D})$-intervals; then we show that, if $T_\Pi^i(\mathfrak{I}_\mathcal{D})$ entails finitely many such facts, so does $T_\Pi^{i+1}(\mathfrak{I}_\mathcal{D})$ and thus eventually also $T_\Pi^k(\mathfrak{I}_\mathcal{D})$, which is precisely $\mathfrak{C}_{\Pi,\mathcal{D}}$.

Assume now that Π and $\mathcal{D}$ entail finitely many relational facts over $(\Pi, \mathcal{D})$-intervals, and so, the same holds for $T_\Pi^0(\mathfrak{I}_\mathcal{D}) = \mathfrak{I}_\mathcal{D}$. We can show that if $T_\Pi^{\alpha+1}(\mathfrak{I}_\mathcal{D}) \neq T_\Pi^\alpha(\mathfrak{I}_\mathcal{D})$, for an ordinal α, then $T_\Pi^{\alpha+1}(\mathfrak{I}_\mathcal{D})$ entails at least one more relational fact over a $(\Pi, \mathcal{D})$-interval than $T_\Pi^\alpha(\mathfrak{I}_\mathcal{D})$. Hence, if there was no finite α such that $T_\Pi^\alpha(\mathfrak{I}_\mathcal{D}) = \mathfrak{C}_{\Pi,\mathcal{D}}$, then Π and $\mathcal{D}$ would entail infinitely many relational facts over $(\Pi, \mathcal{D})$-intervals, in direct contradiction with our assumption. □

For instance, recall that the consequence operator T_Π of program Π in Example 7 does not converge after finitely many steps for the dataset $\mathcal{D}$ given there. Note that, although $\mathfrak{C}_{\Pi,\mathcal{D}}$ can be finitely represented by the fact $P@[0, \infty)$, the interval $[0, \infty)$ contains infinitely many $(\Pi, \mathcal{D})$-intervals and hence, as stated in Lemma 9, Π and $\mathcal{D}$ entail infinitely many facts over $(\Pi, \mathcal{D})$-intervals.

In the remainder of this section, we proceed as follows.

- In Section 4.1 we use Lemma 9 to show that a bounded program Π is finitely materialisable for bounded datasets if and only if T_Π converges in finitely many steps for a specific *critical dataset* $\mathcal{D}_\Pi$ constructed from Π.
- Using the results from Section 4.1, we show in Section 4.2 that finite materialisability of bounded programs for bounded datasets is decidable in ExpTime.
- In Section 4.3, we propose a sufficient condition for finite materialisability in our setting that can be checked in coNP, and even in NL for certain types of programs.

4.1 The Critical Dataset

We next show that finite materialisability for bounded programs and datasets can be solved by focusing on a specific critical dataset. The idea of constructing a critical dataset has been exploited to establish both decidability and undecidability results for a number of reasoning problems, such as deciding universal termination of various chase procedures for extensions of Datalog (Gogacz and Marcinkowski 2014; Cuenca Grau et al. 2013; Marnette and Geerts 2010).

In our setting, however, the temporal domain needs to be taken into account, which makes the definition of the critical dataset more involved. In what follows, for a program Π, we let ℓ_Π be the sum of all rational endpoints of intervals mentioned in Π, or 0 if Π does not mention any such endpoints.

Definition 10. *The* critical dataset $\mathcal{D}_\Pi$ *for a bounded program* Π *is the set of all relational facts of the form* $P(\mathbf{s})@[0, 2\ell_\Pi]$*, where* P *occurs in* Π *and* $\mathbf{s}$ *mentions only constants from* Π *extended with a single fresh constant* c_Π*.*

All facts in the critical dataset $\mathcal{D}_\Pi$ are over the same interval $[0, 2\ell_\Pi]$, and hence $\mathcal{D}_\Pi$ has exponentially many facts in the size of Π. The choice of the interval $[0, 2\ell_\Pi]$ is justified by the technical lemma given next. For an interpretation $\mathfrak{I}$ and an interval ϱ, we define the *projection* $\mathfrak{I} \mid_\varrho$ of $\mathfrak{I}$ over ϱ as the interpretation that coincides with $\mathfrak{I}$ on ϱ and makes all relational atoms false outside ϱ. Then, Lemma 11 shows that if a bounded program Π and a dataset $\mathcal{D}$ entail infinitely many relational facts over $(\Pi, \mathcal{D})$-intervals, then so do Π and any dataset $\mathcal{D}'$ representing a projection of the canonical interpretation $\mathfrak{C}_{\Pi,\mathcal{D}}$ over some interval of length $2\ell_\Pi$. Hence, it suffices to include in the critical dataset facts over any interval of length $2\ell_\Pi$. In what follows, for a dataset $\mathcal{D}$ we let $t_\mathcal{D}^-$ and $t_\mathcal{D}^+$ be the minimal and maximal numbers mentioned as interval endpoints in $\mathcal{D}$, or 0 if there are no such numbers.

Lemma 11. *Let* Π *be a bounded program,* $\mathcal{D}$ *a bounded dataset,* ϱ *a closed interval of length* $2\ell_\Pi$*, and let* $\mathcal{D}'$ *be any dataset representing the projection of* $\mathfrak{C}_{\Pi,\mathcal{D}}$ *over* ϱ*. Then:*

1. *If* $\varrho^+ \geq t_\mathcal{D}^+$*, then* $\mathfrak{C}_{\Pi,\mathcal{D}} \mid_{(\varrho^+,\infty)} = \mathfrak{C}_{\Pi,\mathcal{D}'} \mid_{(\varrho^+,\infty)}$.
2. *If* $\varrho^- \leq t_\mathcal{D}^-$*, then* $\mathfrak{C}_{\Pi,\mathcal{D}} \mid_{(-\infty,\varrho^-)} = \mathfrak{C}_{\Pi,\mathcal{D}'} \mid_{(-\infty,\varrho^-)}$.

Proof sketch. Both statements can be proved analogously. We focus on the first one, and thus assume that

$\varrho^+ \geq t^+_{\mathcal{D}}$. The fact that $\mathfrak{C}_{\Pi,\mathcal{D}'} \restriction_{(\varrho^+,\infty)} \subseteq \mathfrak{C}_{\Pi,\mathcal{D}} \restriction_{(\varrho^+,\infty)}$, holds since $\mathfrak{C}_{\Pi,\mathcal{D}'} \subseteq \mathfrak{C}_{\Pi,\mathcal{D}}$ follows from the definition of $\mathcal{D}'$, which implies $\mathfrak{C}_{\Pi,\mathcal{D}} \models \mathcal{D}'$. To show that $\mathfrak{C}_{\Pi,\mathcal{D}} \restriction_{(\varrho^+,\infty)} \subseteq \mathfrak{C}_{\Pi,\mathcal{D}'} \restriction_{(\varrho^+,\infty)}$ it suffices to prove inductively that, for every ordinal α and relational fact $M@t$ with $t > \varrho^+$, if $T^\alpha_\Pi(\mathfrak{I}_\mathcal{D}) \models M@t$, then $T^\alpha_\Pi(\mathfrak{I}_{\mathcal{D}'}) \models M@t$. The base case holds trivially since $T^0_\Pi(\mathfrak{I}_\mathcal{D}) \not\models M@t$, due to the fact that $T^0_\Pi(\mathfrak{I}_\mathcal{D}) = \mathfrak{I}_\mathcal{D}$ and $t > t^+_\mathcal{D}$. In the inductive step for a successor ordinal $\alpha + 1$, we assume that $T^{\alpha+1}_\Pi(\mathfrak{I}_\mathcal{D}) \models M@t$ but $T^\alpha_\Pi(\mathfrak{I}_\mathcal{D}) \not\models M@t$. Therefore, there is a rule $M' \leftarrow M_1 \wedge \cdots \wedge M_n$ in $\mathsf{ground}(\Pi, \mathcal{D})$ and a time point t' such that $T^\alpha_\Pi(\mathfrak{I}_\mathcal{D}) \models \{M_1@t', \ldots, M_n@t'\}$ and $M'@t' \models M@t$. First, we observe that since $T^\alpha_\Pi(\mathfrak{I}_\mathcal{D}) \models \{M_1@t', \ldots, M_n@t'\}$, there exists $\mathcal{D}''$ such that $T^\alpha_\Pi(\mathfrak{I}_\mathcal{D}) \models \mathcal{D}''$ and $\mathcal{D}'' \models \{M_1@t', \ldots, M_n@t'\}$; moreover, by the definition of ℓ_Π we can show the existence of such $\mathcal{D}''$ which mentions only intervals contained in $[t' - \ell_\Pi, t' + \ell_\Pi]$. Since $M'@t' \models M@t$, the definition of ℓ_Π implies $|t - t'| \leq \ell_\Pi$. Hence, the intervals mentioned in $\mathcal{D}''$ are contained in $[t - 2\ell_\Pi, t + 2\ell_\Pi]$. As $t > \varrho^+$ and $\varrho^+ - \varrho^- \geq 2\ell_\Pi$, these intervals are contained in $[\varrho^-, t + 2\ell_\Pi]$. Thus, by the inductive assumption and $\mathfrak{C}_{\Pi,\mathcal{D}} \restriction_\varrho = \mathfrak{C}_{\Pi,\mathcal{D}'} \restriction_\varrho$, we get $T^{\alpha+1}_\Pi(\mathfrak{I}_{\mathcal{D}'}) \models M@t$, as required. Finally, we note that for a limit ordinal α the inductive step is straightforward since $T^\alpha_\Pi(\mathfrak{I}_\mathcal{D}) = \bigcup_{\beta<\alpha} T^\beta_\Pi(\mathfrak{I}_\mathcal{D})$. □

Using Lemmas 9 and 11, we can now show that checking finite materialisability of Π is tantamount to checking finite convergence of T_Π for the critical dataset associated to Π.

Theorem 12. *A bounded program Π is finitely materialisable for the class of bounded datasets if and only if T_Π converges for $\mathcal{D}_\Pi$ in finitely many steps.*

Proof sketch. The "if" direction holds since $\mathcal{D}_\Pi$ is a bounded dataset. Next, we prove the contrapositive of the "only if" implication. Let $\mathcal{D}$ be a bounded dataset such that T_Π does not finitely converge for $\mathcal{D}$. By Lemma 9, Π and $\mathcal{D}$ entail infinitely many relational facts over $(\Pi, \mathcal{D})$-intervals and, in particular, infinitely many such facts located to the right of $t^+_\mathcal{D}$ or to the left of $t^-_\mathcal{D}$. We assume the former case (the proof for the latter case is symmetric) and show that T_Π does not converge for $\mathcal{D}_\Pi$ in finitely many steps. For this, we successively transform $\mathcal{D}$ such that Π and each of the constructed datasets entail infinitely many facts over ruler-intervals. First, let $\mathcal{D}'$ be a dataset which entails exactly those relational facts $M@t$, for which $\mathfrak{C}_{\Pi,\mathcal{D}} \models M@t$ and $t \in [t^+_\mathcal{D} - 2\ell_\Pi, t^+_\mathcal{D}]$. By Statement 1 from Lemma 11, Π and $\mathcal{D}'$ entail infinitely many relational facts over $(\Pi, \mathcal{D}')$-intervals. If we add to $\mathcal{D}'$ more facts or if we shift all intervals in $\mathcal{D}'$ by the same distance, still infinitely many relational facts over ruler-intervals will be entailed. In particular, this is the case for a dataset $\mathcal{D}''$ consisting of all relational facts $M@[0, 2\ell_\Pi]$ with M occurring in $\mathsf{ground}(\Pi, \mathcal{D})$. Observe that $\mathcal{D}_\Pi$ can be obtained from $\mathcal{D}''$ by replacing each relational atom M with a relational atom $M[c_\Pi]$ where constants not occurring in Π are replaced with a single fresh constant c_Π. We can show inductively that for every ordinal α and for every metric fact $M@t$, if $T^\alpha_\Pi(\mathfrak{I}_{\mathcal{D}''}) \models M@t$, then $T^\alpha_\Pi(\mathfrak{I}_{\mathcal{D}_\Pi}) \models M[c_\Pi]@t$. Hence, Π and $\mathcal{D}_\Pi$ entail infinitely many relational facts over $(\Pi, \mathcal{D}_\Pi)$-intervals. Thus, by Lemma 9, T_Π does not converge for $\mathcal{D}_\Pi$ in any finite number of steps. □

4.2 Complexity Upper Bounds

By Theorem 12, checking if Π is finitely materialisable for bounded datasets reduces to checking whether T_Π finitely converges for the critical dataset $\mathcal{D}_\Pi$. The following lemma suggests a method to check for this. In particular, it shows that a finitely materialisable bounded program Π and a bounded dataset $\mathcal{D}$ cannot entail facts over time points that are "too far away" to the left of $t^-_\mathcal{D}$ or to the right of $t^+_\mathcal{D}$. Hence, if the applications of T_Π to $\mathfrak{I}_\mathcal{D}$ ever yield one such fact, T_Π does not converge for $\mathcal{D}$ in finitely many steps.

Lemma 13. *and let $\mathcal{D}$ be a bounded dataset. If Π is finitely materialisable for the class of bounded datasets, and $(\Pi, \mathcal{D}) \models M@t$ with M a ground relational atom, then $t \in [t^-_\mathcal{D} - 3p_\Pi\ell_\Pi, t^+_\mathcal{D} + 3p_\Pi\ell_\Pi]$, where p_Π is the number of predicates mentioned in Π.*

Proof sketch. Let $\mathcal{D}'$ contain all relational facts $M'@\varrho'$ with M' in $\mathsf{ground}(\Pi, \mathcal{D})$ and ϱ' a $(\Pi, \mathcal{D})$-interval within $[t^-_\mathcal{D}, t^+_\mathcal{D}]$. Since $\mathcal{D}' \models \mathcal{D}$, $t^-_{\mathcal{D}'} = t^-_\mathcal{D}$, and $t^+_{\mathcal{D}'} = t^+_\mathcal{D}$, it suffices to show that $t \in [t^-_\mathcal{D} - 3p_\Pi\ell_\Pi, t^+_\mathcal{D} + 3p_\Pi\ell_\Pi]$, for each $M@t$ entailed by Π and $\mathcal{D}'$. Let $\ldots, \varrho_{-1}, \varrho_0, \varrho_1, \ldots$ be the sequence of consecutive $(\Pi, \mathcal{D}')$-intervals with $\varrho_0 = [t^+_\mathcal{D}, t^+_\mathcal{D}]$, and let $\ldots, \mathcal{R}_{-1}, \mathcal{R}_0, \mathcal{R}_1, \ldots$ be the sequence of sets of the relational facts entailed by Π and $\mathcal{D}'$ over these intervals. Each interval $[t, t + \ell_\Pi)$, with t in the $(\Pi, \mathcal{D}')$-ruler, contains $k = d \cdot \frac{\ell_\Pi}{\mathsf{div}(\Pi)}$ $(\Pi, \mathcal{D}')$-intervals, with d the number of $(\Pi, \mathcal{D}')$-intervals contained in $[t^+_\mathcal{D}, t^+_\mathcal{D} + \mathsf{div}(\Pi))$. Hence, we need to show that $\mathcal{R}_i = \emptyset$ whenever $i < m - 3kp_\Pi$ or $i > 3kp_\Pi$, for m such that $\varrho_m = [t^-_\mathcal{D}, t^-_\mathcal{D}]$. These cases have symmetric proofs, so we focus on the case with $i > 3kp_\Pi$.

We can show inductively that, for each ordinal α, ground relational atom $P(\mathbf{s})$, and $t \geq t^+_\mathcal{D}$: if $T^\alpha_\Pi(\mathfrak{I}_{\mathcal{D}'}) \models P(\mathbf{s})@t$, then $T^\alpha_\Pi(\mathfrak{I}_{\mathcal{D}'}) \models P(\mathbf{s}')@[t^+_\mathcal{D}, t]$, for each tuple $\mathbf{s}'$ of constants from $\mathcal{D}'$ of suitable arity. Hence, $\mathcal{R}_0 \supseteq \mathcal{R}_1 \supseteq \ldots$ and for all $i \geq 0$, if $P(\mathbf{s}) \in \mathcal{R}_i$, then $P(\mathbf{s}') \in \mathcal{R}_i$ for each $\mathbf{s}'$. The latter implies that $\mathcal{R}_0, \mathcal{R}_1, \ldots$ has at most p_Π distinct non-empty sets, and hence we need to show that there is no $i \geq 0$ such that $\mathcal{R}_i = \mathcal{R}_{i+1} = \cdots = \mathcal{R}_{i+3k}$ is a sequence of non-empty sets. Towards a contradiction assume there is such i. If ϱ_i is punctual, we let $j = i$ and otherwise $j = i + 1$. By the definition, $k \geq 2$, so $j + 2k + 1 \leq i + 3k$, and thus $\mathcal{R}_j = \cdots = \mathcal{R}_{j+2k+1}$. Now, for each $i \in \mathbb{Z}$ we let $\mathcal{D}_i$ be the set of relational facts $M@\varrho_i$ such that $M \in \mathcal{R}_i$. Observe that $\varrho = \varrho_j \cup \cdots \cup \varrho_{j+2k}$ is a closed interval of length $2\ell_\Pi$ and $\varrho^+ \geq t^+_{\mathcal{D}_\Pi}$. By Lemma 11, Π and $\mathcal{D}_j \cup \cdots \cup \mathcal{D}_{j+2k}$ entail $\mathcal{D}_{j+2k+1}$. Since $\mathcal{R}_j = \cdots = \mathcal{R}_{j+2k+1}$, further applications of Lemma 11 allow us to show that $\mathcal{R}_{j+2k+1} = \mathcal{R}_{j+2k+2} = \ldots$ is an infinite sequence of sets. Since these sets are non-empty, Π and $\mathcal{D}'$ entail infinitely many relational facts over $(\Pi, \mathcal{D}')$-intervals, so Lemma 9 yields a contradiction. □

Algorithm 2 provides a decision procedure for finite materialisability by exploiting the results in Theorem 12 and

Algorithm 2: Checking finite materialisability

Input: A bounded program Π
Output: A truth value

1 Construct $\mathcal{D}_\Pi$ and assign $\mathcal{D}' := \mathcal{D}_\Pi$;
2 Assign $\varrho = [-3p_\Pi \ell_\Pi, (3p_\Pi + 2)\ell_\Pi]$;
3 **loop**
4 | Assign $\mathcal{D}_{aux} := \mathcal{D}'$;
5 | Assign $\mathcal{D}' :=$ the dataset representing $T_\Pi(\mathfrak{I}_{\mathcal{D}'})$ obtained by the algorithm from Theorem 4 ;
6 | **if** *there is* $M@\varrho' \in \mathcal{D}'$ *with* $\varrho' \not\subseteq \varrho$ **then**
7 | | Return false
8 | **else if** $\mathcal{D}_{aux} \models \mathcal{D}'$ **then**
9 | | Return true

Lemma 13. On input Π, the algorithm first constructs the critical dataset $\mathcal{D}_\Pi$ and computes the instantiation ϱ of the interval mentioned in Lemma 13. It then proceeds, similarly to Algorithm 1 in Section 3, by iteratively computing datasets representing the successive applications of T_Π until either a fact outside ϱ is derived (in which case false is returned) or otherwise until a fixpoint is reached (in which case true is returned). Correctness follows from Theorem 12 and Lemma 13; furthermore, Algorithm 2 terminates since, eventually, either the dataset $\mathcal{D}'$ computed within the loop will entail all derivable facts over ϱ or it will contain a fact involving an interval not in ϱ. A more detailed analysis also reveals that Algorithm 2 can be made work in exponential time, thus obtaining the result in the following theorem.

Theorem 14. *Checking whether a bounded program is finitely materialisable for bounded datasets is in* ExpTime.

Proof sketch. The procedure uses Algorithm 2, whose correctness follows from Theorem 12 and Lemma 13. We next argue that Algorithm 2 terminates in exponential time. The algorithm constructs $\mathcal{D}_\Pi$ and ϱ in Lines 1 and 2, which is clearly feasible in exponential time. We argue that an iteration of the loop in Lines 3–9 takes at most exponential time and computes $\mathcal{D}'$ of exponential size. Consider the ith iteration of the loop, where $\mathcal{D}'$ represents $T^i_\Pi(\mathfrak{I}_{\mathcal{D}_\Pi})$, and was constructed by applying i times the algorithm from Theorem 4. An analysis of this algorithm shows that, for each $M@\varrho' \in \mathcal{D}'$, atom M occurs in $\mathsf{ground}(\Pi, \mathcal{D}_\Pi)$ and the endpoints of ϱ' belong to the $(\Pi, \mathcal{D}_\Pi)$-ruler. Moreover, $\varrho' \subseteq \varrho$ since otherwise Algorithm 2 would have stopped in the previous loop iteration. Hence, the number of facts in $\mathcal{D}'$ is bounded by $A \cdot B$, with A the number of atoms mentioned in $\mathsf{ground}(\Pi, \mathcal{D}_\Pi)$ and B the number of $(\Pi, \mathcal{D}_\Pi)$-intervals contained in ϱ. Both A and B are of exponential size since $A = p_\Pi \cdot (con + 1)^{ar}$, where con is the number of constants in Π and ar is the maximum predicate arity in Π, whereas $B = 1 + \frac{2(6+2p_\Pi)\ell_\Pi}{\mathsf{div}(\Pi)}$. Since $\mathcal{D}'$ is exponentially big, we can show that the algorithm from Theorem 4 constructs in exponential time a dataset representing $T_\Pi(\mathfrak{I}_{\mathcal{D}'})$ in Line 5. The condition in Line 6 can be checked in exponential time, and so can be the condition in Line 8, by checking if for each $M@\varrho' \in \mathcal{D}'$, there exist $M@\varrho_1, \ldots, M@\varrho_k$ in $\mathcal{D}_{aux}$ such that $\varrho' \subseteq \varrho_1 \cup \cdots \cup \varrho_k$. Finally, the main loop goes through exponentially many iterations: the operation in Line 5 extends $\mathcal{D}'$ to a dataset entailing additional relational facts over $(\Pi, \mathcal{D}_\Pi)$-intervals, and there are at most $A \cdot B$ such facts. □

4.3 Sufficient Conditions

We next propose a sufficient condition for finite materialisability of bounded programs. Our condition is based on generalising the notion of a program's dependency graph (see Section 2) to *metric dependency graph*, where edges representing dependencies between predicates are labelled with intervals obtained from the program. Acyclicity of the dependency graph used to define non-recursive programs is then generalised to allow for certain cycles. The construction of the metric dependency graph for program Π proceeds in two steps. First, we transform Π into a propositional program by replacing each occurrence of a predicate $P(\mathbf{s})$ with a propositional letter X_P and *strengthen* it by rewriting away metric operators other than $\oplusdot$ and $\ominusdot$ in rule bodies. Second, we construct a graph for the transformed program, where intervals labelling edges are derived from its rules.

We start by defining the strengthening transformation.

Definition 15. *The* strengthening $\mathsf{str}(\Pi)$ *of a ground program* Π *is obtained from* Π *by exhaustively performing the following replacements in rule bodies, for any interval* ϱ *and metric atoms* M_1 *and* M_2*: (i) each occurrence of* $\boxminus_\varrho$ *and* $\boxplus_\varrho$ *is replaced with* $\ominusdot_\varrho$ *and* $\oplusdot_\varrho$ *respectively, (ii) each occurrence of* $M_1 \mathcal{S}_\varrho M_2$ *is replaced with* $\ominusdot_\varrho M_2$*, and (iii) each occurrence of* $M_1 \mathcal{U}_\varrho M_2$ *is replaced with* $\oplusdot_\varrho M_2$*.*

It is straightforward to see that the order in which the replacement rules (i)–(iii) are applied is immaterial to the result of the transformation, and hence $\mathsf{str}(\Pi)$ is well defined. The following proposition shows that the transformation indeed results in a logical strengthening of the program.

Proposition 16. *Let* Π *be a ground program, let* $\mathcal{D}$ *be a dataset, and let* $M@\varrho$ *be a metric fact. If* $(\Pi, \mathcal{D}) \models M@\varrho$*, then* $(\mathsf{str}(\Pi), \mathcal{D}) \models M@\varrho$*.*

Proof. The strengthening transformation leaves rule heads unaltered. We argue that if the body of a rule $r \in \Pi$ holds at some time point t, then the body of the corresponding rule in $\mathsf{str}(\Pi)$ also holds at t. Indeed, simply observe that, by the semantics of metric operators, $\boxminus_\varrho M \models \ominusdot_\varrho M$, $\boxplus_\varrho M \models \oplusdot_\varrho M$, $M_1 \mathcal{S}_\varrho M_2 \models \ominusdot_\varrho M_2$, and $M_1 \mathcal{U}_\varrho M_2 \models \oplusdot_\varrho M_2$. □

We are ready to define metric dependency graphs. To determine the intervals labelling edges we introduce the operations $\varrho + \varrho' = \{t + t' \mid t \in \varrho \text{ and } t' \in \varrho'\}$ and $-\varrho = \{-t \mid t \in \varrho\}$, for ϱ and ϱ' arbitrary intervals.

Definition 17. *Let* Π *be a bounded program and* Π' *be obtained form* Π *by replacing each occurrence of a relational atom* $P(\mathbf{s})$ *with a proposition* X_P *uniquely associated to* P*.*

The metric dependency graph *of* Π *is the edge-labelled directed multigraph* G_Π *with the nodes and edges given next:*

- *A vertex* v_P *for each proposition* X_P *in* $\mathsf{str}(\Pi')$*.*
- *An edge* (v_P, v_Q) *if there is a rule in* $\mathsf{str}(\Pi')$ *where* X_Q *occurs in the head* M' *and* X_P *occurs in a body atom* M*. Let* $\otimes^i_{\varrho_i}$ *for* $1 \leq i \leq k$*, be all occurrences of metric*

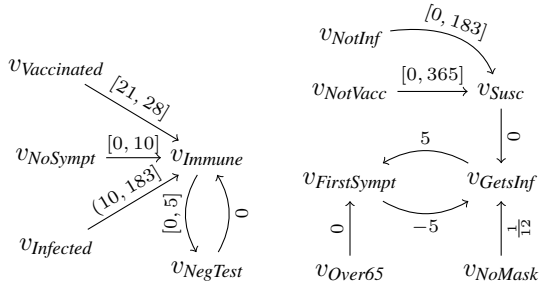

Figure 1: Metric dependency graphs for the programs from Example 3 (a) and Example 20 (b)

operators in M' and let $\otimes^i_{\varrho_i}$ for $k+1 \leq i \leq k+m$ be all occurrences of metric operators in M; then, the edge is labelled with $\varrho = [0,0] + \check{\varrho}_1 + \cdots + \check{\varrho}_{k+m}$, where

$$\check{\varrho}_i = \begin{cases} \varrho_i & \text{if } i \leq k \text{ and } \otimes^i = \boxplus, \text{ or } i > k \text{ and } \otimes^i = \diamondminus, \\ -\varrho_i & \text{if } i \leq k \text{ and } \otimes^i = \boxminus, \text{ or } i > k \text{ and } \otimes^i = \diamondplus. \end{cases}$$

The interval weight of a path *in G_Π is $[0,0] + \varrho_1 + \cdots + \varrho_n$ where $\varrho_1, \ldots, \varrho_n$ are intervals labelling the edges of the path.*

Note that the interval weight of a finite path in G_Π is a bounded interval since each edge label is bounded and the interval weight of a path with a single vertex is $[0,0]$. The metric dependency graph of the program from Example 3 is depicted in Figure 1(a).

The following lemma establishes the key connection between facts that are entailed by a program Π and a dataset $\mathcal{D}$ and reachability in the metric dependency graph G_Π.

Lemma 18. *Let Π be a bounded program, $\mathcal{D}$ a bounded dataset, and $P'(\mathbf{s}')@t'$ a relational fact. If $(\Pi, \mathcal{D}) \models P'(\mathbf{s}')@t'$, there is $P(\mathbf{s})@\varrho \in \mathcal{D}$ and a path from v_P to $v_{P'}$ in G_Π with interval weight ϱ' such that $t' \in \varrho + \varrho'$.*

Proof sketch. We show, for each ordinal α, that $T^\alpha_\Pi(\mathcal{D}) \models P'(\mathbf{s}')@t'$ implies the existence of $P(\mathbf{s})@\varrho \in \mathcal{D}$ and a path in G_Π from v_P to $v_{P'}$ with interval weight ϱ' such that $t' \in \varrho + \varrho'$. The proof is inductive on α. In the base case $T^0_\Pi(\mathcal{D}) = \mathfrak{I}_\mathcal{D}$, so $T^0_\Pi(\mathcal{D}) \models P'(\mathbf{s}')@t'$ implies that there is a fact $P'(\mathbf{s}')@\varrho \in \mathcal{D}$ with $t' \in \varrho$. The required path in G_Π consists of a single vertex $v_{P'}$: its weight is $[0,0]$ and $t' \in \varrho + [0,0]$. For the inductive step, assume $T^{\alpha+1}_\Pi(\mathcal{D}) \models P'(\mathbf{s}')@t'$ and $T^\alpha_\Pi(\mathcal{D}) \not\models P'(\mathbf{s}')@t'$. Hence, there exist $r \in \mathsf{ground}(\Pi)$ and t such that the body of r holds at t in $T^\alpha_\Pi(\mathcal{D})$, and the head holding at t entails $P'(\mathbf{s}')@t'$. By the proof of Proposition 16, this also holds for the corresponding rule $M' \leftarrow M_1 \wedge \cdots \wedge M_n$ in $\mathsf{str}(\{r\})$, where M' is of the form $\otimes^1_{\varrho_1} \cdots \otimes^k_{\varrho_k} P'(\mathbf{s}')$ and M_1 is of the form $\otimes^{k+1}_{\varrho_{k+1}} \cdots \otimes^{k+m}_{\varrho_{k+m}} P_1(\mathbf{s}_1)$ for $P_1(\mathbf{s}_1)$ a relational atom, $\otimes^1 \cdots \otimes^k$ a sequence of operators $\boxminus$ and $\boxplus$, and $\otimes^{k+1} \cdots \otimes^{k+m}$ a sequence of operators $\diamondminus$ and $\diamondplus$. By Definition 17, G_Π has an edge e from v_{P_1} to $v_{P'}$, labelled with $[0,0] + \check{\varrho}_1 + \cdots + \check{\varrho}_{k+m}$, and we can show that $t' - t \in [0,0] + \check{\varrho}_1 + \cdots + \check{\varrho}_k$. Now, since $T^\alpha_\Pi(\mathcal{D}) \models \otimes^{k+1}_{\varrho_{k+1}} \cdots \otimes^{k+m}_{\varrho_{k+m}} P_1(\mathbf{s}_1)@t$, there is t_1 such that $T^\alpha_\Pi(\mathcal{D}) \models P_1(\mathbf{s}_1)@t_1$ and $t - t_1 \in [0,0] + \check{\varrho}_{k+1} + \cdots + \check{\varrho}_{k+m}$. By the inductive assumption, there exist $P(\mathbf{s})@\varrho \in \mathcal{D}$ and a path in G_Π from v_P to v_{P_1} with interval weight ϱ' such that $t_1 \in \varrho + \varrho'$. Extending this path with an edge e yields a path with weight $\varrho'' = \varrho' + \check{\varrho}_1 + \cdots + \check{\varrho}_{k+m}$, and we can show that $t' \in \varrho + \varrho''$. The inductive step for a limit ordinal α is straightforward since $T^\alpha_\Pi(\mathcal{D}) = \bigcup_{\beta<\alpha} T^\beta_\Pi(\mathcal{D})$. □

By Lemma 18, if Π and $\mathcal{D}$ entail facts $P'(\mathbf{s}')@t'$ for arbitrarily large/small t' (thus, Π is not finitely materialisable), there must be a cycle in G_Π with non-zero interval weight. Otherwise, t' could not have been arbitrarily large/small. Our sufficient condition is based on this observation.

Definition 19. *A bounded program Π is MTL-acyclic if each cycle in G_Π has interval weight $[0,0]$.*

Non-recursive bounded DatalogMTL programs as well as all plain Datalog programs are trivially MTL-acyclic. Furthermore, so are programs exhibiting "safe" temporal recursion, as shown by the following example.

Example 20. *It is believed that individuals who have been neither vaccinated against COVID-19 in the last year nor infected for the last half a year are susceptible to COVID-19. Usually, a susceptible individual who takes off their mask and remains in contact with an infected person for 2 hours, gets infected. Individuals over 65 develop first symptoms 5 days after getting infected. It is also estimated that for individuals who develop first symptoms of COVID-19, the source of infection should be looked for on the 5th day preceding the current one. A* DatalogMTL *program representing these dependencies consists of the following rules:*

$$\begin{aligned} \mathit{Susc}(x) &\leftarrow \boxminus_{[0,365]} \mathit{NotVacc}(x) \wedge \boxminus_{[0,183]} \mathit{NotInf}(x), \\ \mathit{GetsInf}(x) &\leftarrow \mathit{ContInf}(x,y) \mathcal{S}_{\frac{1}{12}} \mathit{NoMask}(x) \wedge \mathit{Susc}(x), \\ \mathit{FirstSympt}(x) &\leftarrow \boxminus_5 \mathit{GetsInf}(x) \wedge \mathit{Over65}(x), \\ \boxminus_5 \mathit{GetsInf}(x) &\leftarrow \mathit{FirstSympt}(x). \end{aligned}$$

The metric dependency graph of this program is given in Figure 1(b). It has one simple cycle and its weight is $[0,0]$, so the program is MTL-acyclic.

Now, we show that MTL-acyclicity provides a sufficient condition for finite materialisability.

Theorem 21. *Bounded MTL-acyclic programs are finitely materialisable for the class of bounded datasets.*

Proof. We show the contrapositive. Let Π be a bounded program and $\mathcal{D}$ a bounded dataset such that T_Π does not converge for $\mathcal{D}$ in finitely many steps. By Lemma 9, Π and $\mathcal{D}$ entail infinitely many relational facts over $(\Pi, \mathcal{D})$-intervals. Let ϱ_Π be the sum of all labels in G_Π (or [0,0] if G_Π has no edges). Thus, there exists a fact $P'(\mathbf{s}')@t'$ with $t' \notin [t^-_\mathcal{D} - \varrho^-_\Pi, t^+_\mathcal{D} + \varrho^+_\Pi]$ such that $(\Pi, \mathcal{D}) \models P'(\mathbf{s}')@t'$. We will focus on the case $t' > t^+_\mathcal{D} + \varrho^+_\Pi$ (the case $t' < t^-_\mathcal{D} - \varrho^-_\Pi$ has an analogous proof). Since $(\Pi, \mathcal{D}) \models P'(\mathbf{s}')@t'$, by Lemma 18, there are $P(\mathbf{s})@\varrho \in \mathcal{D}$ and a path σ in G_Π from v_P to $v_{P'}$ with interval weight ϱ' such that $t' \in \varrho + \varrho'$. As

$t' > t^+_{\mathcal{D}} + \varrho^+_\Pi$ and $t^+_{\mathcal{D}} \geq \varrho^+$, we have $t' > \varrho^+ + \varrho^+_\Pi$. On the other hand, $t' \in \varrho + \varrho'$ implies $\varrho^+ + \varrho'^+ \geq t'$, and so, $\varrho'^+ > \varrho^+_\Pi$. Let now σ' be any path obtained from σ by deleting all cycles whose interval weights are $[0, 0]$; note that removing such cycles does not change the interval weight ϱ' of the path. However, since $\varrho'^+ > \varrho^+_\Pi$, there needs to be a cycle in σ'. By the construction, the interval weight of this cycle is not $[0, 0]$, so Π is not MTL-acyclic. □

Observe, however, that not every finitely materialisable program is MTL-acyclic. For instance, this is the case for the program presented in Example 3. Indeed, the metric dependency graph of this program has a cycle with non-zero weight $[0, 5]$ as depicted in Figure 1(a).

We next analyse the complexity of checking whether a program is MTL-acyclic, and show that the check is feasible in coNP. Additionally, the complexity drops to NL for *forward propagating* programs (Wałęga et al. 2019), where $\diamondminus$, $\boxminus$, and $\mathcal{S}$ are the only metric operators allowed in rule bodies and $\boxplus$ the only operator allowed in heads. The same holds for the *backwards propagating* programs, where $\diamondplus$, $\boxplus$, and $\mathcal{U}$ are allowed in rule bodies and only $\boxminus$ is allowed in heads.

Theorem 22. *Checking whether a bounded program is MTL-acyclic is feasible in* coNP*, and in* NL *if the program is additionally either forward or backwards propagating.*

Proof. Let Π be bounded. If Π is not MTL-acyclic, then G_Π has a cycle, and in particular a simple cycle, with interval weight distinct from $[0, 0]$. To check if Π is not MTL-acyclic we construct G_Π, guess its path, and verify that it is a cycle with interval weight different from $[0, 0]$. We argue that this is feasible in NP. Clearly, G_Π can be constructed in polynomial time and its simple cycle can be guessed in NP. Verifying that the guessed path is a cycle with interval weight distinct from $[0, 0]$ is also feasible in polynomial time since, essentially, it amounts to adding rational numbers given in binary. Hence, checking if Π is MTL-acyclic is in coNP. If Π is forward propagating, then all intervals labelling edges of G_Π are positive. Hence, Π is not MTL-acyclic if G_Π has a simple cycle with at least one edge labelled with an interval distinct from $[0, 0]$. Such a path can be guessed and verified vertex-by-vertex in NL. Hence, by coNL = NL, checking if Π is MTL-acyclic is in NL. If Π is backwards propagating, intervals labelling edges of G_Π are negative. Thus, checking if Π is MTL-acyclic can be done analogously to the check for forward propagating programs, and so, it is in NL. □

Interestingly, our condition in Definition 19 characterises finite materialisability for a propositional DatalogMTL$^\Diamond_{\mathsf{core}}$ fragment of bounded DatalogMTL (Wałęga et al. 2020b), where each rule contains a single body atom and no operators other than $\diamondminus$ and $\diamondplus$ are allowed in rules.

Theorem 23. *A bounded propositional* DatalogMTL$^\Diamond_{\mathsf{core}}$ *program is finitely materialisable for the class of bounded datasets if and only if it is MTL-acyclic.*

Proof. The forward direction follows directly from Theorem 21. For the other direction let Π be a a bounded propositional core program which is not MTL-acyclic. Hence, G_Π has a cycle σ whose interval weight is not $[0, 0]$. We can show that if G_Π has an edge from v_P to $v_{P'}$ with label ϱ, then, for any time point t, we have $(\Pi, \{P@t\}) \models P'@t+\varrho$. Indeed, if there is such an edge, then Π needs to have a rule $P' \leftarrow \otimes^1_{\varrho_1} \cdots \otimes^m_{\varrho_m} P$ such that $\otimes^1 \cdots \otimes^m$ is a sequence of the operators $\diamondminus$ and $\diamondplus$, whereas $\varrho = [0, 0] + \check{\varrho}_1 + \cdots + \check{\varrho}_m$. An application of this rule to $P@t$ derives $P'@t + \varrho$. Next, let ϱ_σ be the interval weight of σ, let v_Q be a vertex in σ, and let $\mathcal{D} = \{Q@0\}$. By the previous statement, $(\Pi, \mathcal{D}) \models Q@0 + \varrho_\sigma$. Since ϱ_σ is not $[0, 0]$, there is some $t_\sigma \in \varrho_\sigma$ distinct from 0 such that $(\Pi, \mathcal{D}) \models Q@t_\sigma$. Thus, for any integer $i \geq 0$, we have $(\Pi, \mathcal{D}) \models Q@i \cdot t_\sigma$, so Π and $\mathcal{D}$ entail infinitely many relational facts over $(\Pi, \mathcal{D})$-intervals; we can now apply Lemma 9 to complete the proof. □

5 Fact Entailment

We now focus on the complexity of reasoning over finitely materialisable bounded programs. In particular, we show that fact entailment in this setting is ExpTime-complete (in combined complexity), and so no harder than fact entailment for plain Datalog. The upper bound is obtained from Algorithm 1 which, by Lemma 13, terminates on finitely materialisable programs in exponentially many steps.

Theorem 24. *Checking whether a finitely materialisable bounded program and a bounded dataset entail a fact over a bounded interval is* ExpTime*-complete.*

Proof. The lower bound follows by a straightforward reduction from fact entailment in plain Datalog, which is ExpTime-complete (Dantsin et al. 2001). For the upper bound, assume that we want to check if a bounded program Π, which is finitely materialisable for bounded datasets, and a bounded dataset $\mathcal{D}$ entail a relational fact $M@\varrho$, with some bounded ϱ. To this end, we can use Algorithm 1 to construct a dataset $\mathcal{D}'$ representing the canonical interpretation $\mathfrak{C}_{\Pi,\mathcal{D}}$ and check if $\mathcal{D}' \models M@\varrho$. By Lemma 13, all relational facts entailed by $\mathfrak{C}_{\Pi,\mathcal{D}}$ lie within $[t^-_{\mathcal{D}} - 3p_\Pi \ell_\Pi, t^+_{\mathcal{D}} + 3p_\Pi \ell_\Pi]$. Since this interval contains exponentially many $(\Pi, \mathcal{D})$-intervals, we can use the same arguments as in the proof of Theorem 14 to show that the algorithm constructs $\mathcal{D}'$ in exponential time. Finally, we observe that $\mathcal{D}' \models M@\varrho$ if and only if there are facts $M@\varrho_1, \ldots, M@\varrho_k$ in $\mathcal{D}'$ such that $\varrho \subseteq \varrho_1 \cup \cdots \cup \varrho_k$. Clearly, the latter can be checked in exponential time. □

6 Conclusions and Future Work

We have proposed and studied a class of DatalogMTL programs that are naturally amenable to materialisation-based reasoning via scalable forward chaining techniques. We see many avenues for future work. From a theoretical standpoint we aim to extend our results to (i) include also unbounded datasets, (ii) study data complexity for reasoning over finitely materialisable programs, and (iii) consider other fragments of DatalogMTL identified in the literature. From a practical perspective, we plan to implement our materialisation techniques for DatalogMTL and combine them with automata-based approaches known from the literature.

Acknowledgments

This work was supported by the EPSRC projects OASIS (EP/S032347/1), AnaLOG (EP/P025943/1), and UK FIRES (EP/S019111/1), the SIRIUS Centre for Scalable Data Access, and Samsung Research UK.

References

Abiteboul, S.; Hull, R.; and Vianu, V. 1995. *Foundations of Databases*. Addison-Wesley.

Artale, A.; Kontchakov, R.; Kovtunova, A.; Ryzhikov, V.; Wolter, F.; and Zakharyaschev, M. 2017. Ontology-mediated query answering over temporal data: A survey. In *Proc. of TIME*.

Baget, J.; Garreau, F.; Mugnier, M.; and Rocher, S. 2014. Extending acyclicity notions for existential rules. In Schaub, T.; Friedrich, G.; and O'Sullivan, B., eds., *Proc. of ECAI*, 39–44.

Brandt, S.; Kontchakov, R.; Ryzhikov, V.; Xiao, G.; and Zakharyaschev, M. 2017. Ontology-based data access with a Horn fragment of metric temporal logic. In *Proc. of AAAI*, 1070–1076.

Brandt, S.; Kalaycı, E. G.; Ryzhikov, V.; Xiao, G.; and Zakharyaschev, M. 2018. Querying log data with metric temporal logic. *J. Artif. Intell. Res.* 62:829–877.

Chomicki, J., and Imieliński, T. 1988. Temporal deductive databases and infinite objects. In *Proc. of PODS*, 61–73.

Chomicki, J. 1990. Polynomial time query processing in temporal deductive databases. In *Proc. of PODS*, 379–391.

Chomicki, J. 1995. Depth-bounded bottom-up evaluation of logic programs. *J. Log. Program.* 25(1):1–31.

Cosmadakis, S. S.; Gaifman, H.; Kanellakis, P. C.; and Vardi, M. Y. 1988. Decidable optimization problems for database logic programs (Preliminary report). In *Proc. of STOC*, 477–490.

Cuenca Grau, B.; Horrocks, I.; Krötzsch, M.; Kupke, C.; Magka, D.; Motik, B.; and Wang, Z. 2013. Acyclicity notions for existential rules and their application to query answering in ontologies. *J. Artif. Intell. Res.* 47:741–808.

Dantsin, E.; Eiter, T.; Gottlob, G.; and Voronkov, A. 2001. Complexity and expressive power of logic programming. *ACM Comput. Surv.* 33(3):374–425.

Gogacz, T., and Marcinkowski, J. 2014. All-instances termination of chase is undecidable. In *Proc. of ICALP Part II*, 293–304.

Grahne, G., and Onet, A. 2018. Anatomy of the chase. *Fundam. Informaticae* 157(3):221–270.

Gutiérrez-Basulto, V.; Jung, J. C.; and Kontchakov, R. 2016a. On decidability and tractability of querying in temporal EL. In *Proc. of DL*.

Gutiérrez-Basulto, V.; Jung, J. C.; and Kontchakov, R. 2016b. Temporalized EL ontologies for accessing temporal data: Complexity of atomic queries. In *Proc. of IJCAI*, 1102–1108.

Kikot, S.; Ryzhikov, V.; Wałęga, P. A.; and Zakharyaschev, M. 2018. On the data complexity of ontology-mediated queries with MTL operators over timed words. In *Proc. of DL*.

Krötzsch, M.; Marx, M.; and Rudolph, S. 2019. The power of the terminating chase (invited talk). In *Proc. of ICDT*, 3:1–3:17.

Marnette, B., and Geerts, F. 2010. Static analysis of schema-mappings ensuring oblivious termination. In *Proc. of ICDT*, 183–195.

Motik, B.; Nenov, Y.; Piro, R.; and Horrocks, I. 2019. Maintenance of datalog materialisations revisited. *Artif. Intell.* 269:76–136.

Ryzhikov, V.; Wałęga, P. A.; and Zakharyaschev, M. 2019. Data complexity and rewritability of ontology-mediated queries in metric temporal logic under the event-based semantics. In *Proc. of IJCAI*, 1851–1857.

Vrandecic, D., and Krötzsch, M. 2014. Wikidata: A free collaborative knowledgebase. *Commun. ACM* 57(10):78–85.

Wałęga, P. A.; Cuenca Grau, B.; Kaminski, M.; and Kostylev, E. V. 2019. DatalogMTL: Computational complexity and expressive power. In *Proc. of IJCAI*, 1886–1892.

Wałęga, P. A.; Cuenca Grau, B.; Kaminski, M.; and Kostylev, E. V. 2020a. DatalogMTL over the integer timeline. In *Proc. of KR*, 768–777.

Wałęga, P. A.; Cuenca Grau, B.; Kaminski, M.; and Kostylev, E. V. 2020b. Tractable fragments of datalog with metric temporal operators. In *Proc. of IJCAI*, 1919–1925.

Wałęga, P.; Cuenca Grau, B.; and Kaminski, M. 2019. Reasoning over streaming data in metric temporal datalog. In *Proc. of AAAI*, 1941–1948.