The Stable Model Semantics of Datalog with Metric Temporal Operators∗†

PRZEMYSŁAW A. WAŁĘGA, DAVID J. TENA CUCALA and BERNARDO CUENCA GRAU

Department of Computer Science, University of Oxford, Oxford OX1 3AZ, UK
(*e-mails:* przemyslaw.walega@cs.ox.ac.uk, david.tena.cucala@cs.ox.ac.uk, bernardo.cuenca.grau@cs.ox.ac.uk)

EGOR V. KOSTYLEV

Department of Informatics, University of Oslo, Oslo, Norway
(*e-mail:* egork@ifi.uio.no)

Abstract

We introduce negation under the stable model semantics in DatalogMTL – a temporal extension of Datalog with metric temporal operators. As a result, we obtain a rule language which combines the power of answer set programming with the temporal dimension provided by metric operators. We show that, in this setting, reasoning becomes undecidable over the rational timeline, and decidable in EXPSPACE in data complexity over the integer timeline. We also show that, if we restrict our attention to forward-propagating programs, reasoning over the integer timeline becomes PSPACE-complete in data complexity, and hence, no harder than over positive programs; however, reasoning over the rational timeline in this fragment remains undecidable.

KEYWORDS: temporal reasoning, metric temporal logic, stable model semantics, non-monotonic negation

1 Introduction

DatalogMTL (Brandt *et al.* 2018) extends positive Datalog (Abiteboul *et al.* 1995) with operators from metric temporal logic (MTL) (Koymans 1990) interpreted over the rational or the integer timeline. For example, the following DatalogMTL rule can be used to state that a bus driver should not work for more than six months (i.e., half a year) in a row:

$$OnLeave(x) \leftarrow BusDriver(x) \land \boxminus_{[0,0.5]} Working(x),$$

---

where the expression $\boxminus_{[0,0.5]} Working(x)$ holds at time t if $Working(x)$ holds continuously in the time interval $[t-0.5, t]$. Some other examples of expressions allowed in DatalogMTL are $\Diamondminus_{[t_1,t_2]}\varphi$, which holds at time t if φ holds at some instant within the time interval $[t-t_2, t-t_1]$, and $\boxplus_{[t_1,t_2]}\varphi$, which uses the "future-oriented" version of the operator $\boxminus$ and holds at time t if φ holds continuously in the time interval $[t+t_1, t+t_2]$. A DatalogMTL dataset consists of facts involving intervals, such as $Working(alex)@[2022, 2023]$, stating that Alex was working continuously in the time interval $[2022, 2023]$. DatalogMTL is thus a very expressive language which allows a user to capture complex definitions, regulations, or events involving temporal intervals. DatalogMTL is powerful enough to capture prominent temporal extensions of Datalog such as Datalog$_{1S}$ (Chomicki and Imieliński 1988; 1989) and Templog (Abadi and Manna 1989), and it has found applications in areas such as ontology-based data access (Brandt *et al.* 2018), stream reasoning (Wałęga *et al.* 2019), and similar ideas were explored in logic programming (Brzoska 1998). Reasoning in DatalogMTL is known to be PSPACE-complete in data complexity over both the rational (Wałęga *et al.* 2019) and the integer timeline (Wałęga *et al.* 2020a).

Motivated by a range of applications, there has recently been a growing interest in logics that combine non-monotonic negation with temporal constructs (Cabalar and Vega 2007; Aguado *et al.* 2013; Cabalar *et al.* 2018; 2020; Beck *et al.* 2018; Zaniolo 2012). Recently, we have proposed an extension of DatalogMTL over the rationals with stratified negation-as-failure, where rules can have negated atoms in the body, but there can be no recursion involving predicates mentioned in such atoms. With such an extension of DatalogMTL, we can express, for example, a bus company's policy that any vehicle older than 12 years must be decommissioned permanently unless it has passed a special inspection in the last year:

$$\boxplus_{[0,\infty)}Decommis(x) \leftarrow \Diamondminus_{(12,\infty)}Manufactured(x) \wedge \mathsf{not}\, \Diamondminus_{[0,1]}PassInspect(x).$$

We also showed that the additional expressive power provided by this type of negation does not increase the complexity of reasoning regardless of whether we consider the rational or the integer timeline (Tena Cucala *et al.* 2021). The restriction to stratifiable programs, however, significantly limits the applicability DatalogMTL to certain types of use cases.

In this paper we take a further step in this direction and consider DatalogMTL equipped with non-stratifiable negation interpreted under the stable model semantics (Gelfond and Lifschitz 1988; Brooks *et al.* 2007; Nogueira *et al.* 2001). This extension paves the way for the use of DatalogMTL in applications where derived information can be retracted in light of new evidence, minimality of models is required, or temporal inertia rules need to be formalised. For instance, consider a bus company with the policy that vehicles that have been serviced at a given time t are automatically booked for a routine service in a year's time (i.e., at time $t+1$, represented by metric operator $\boxplus_{[1,1]}$), but this appointment must be cancelled if the bus is serviced again before then – that is, within the interval $(t+0, t+1)$, represented by $\Diamondplus_{(0,1)}$. This policy can be written using the rule

$$\boxplus_{[1,1]}Service(x) \leftarrow Service(x) \wedge \mathsf{not}\, \Diamondplus_{(0,1)}Service(x),$$

which is not stratifiable as it involves recursion via negation.

Our setting is closely related to the recent research on combining answer set programming (ASP) with temporal logics. For example, temporal equilibrium logic (TEL)

(Cabalar and Vega 2007; Aguado *et al.* 2013; Cabalar *et al.* 2018) combines ASP with linear temporal logic, and LARS combines ASP with window-based temporal constructs for stream reasoning (Beck *et al.* 2018). The logic recently proposed by Cabalar *et al.* (2020) is perhaps the closest to our work, as it combines stable model semantics with propositional MTL interpreted over the natural numbers; this logic, however, is rather different from DatalogMTL, where the use of logical connectives and MTL operators is restricted in the spirit of Datalog to disallow disjunction and "existential" MTL operators (such as *diamond*, *since*, or *until* operators) in rule heads. As we show in our paper, considering a language with such restrictions can lead to favourable computational behaviour.

Our contributions in this paper are summarised as follows. In Section 3 we present our extension (DatalogMTL$^-$) of DatalogMTL with negation under stable model semantics. Our language is defined similarly to other temporal ASP formalisms; it extends both DatalogMTL with stratified negation and Datalog with stable model negation. To capture the semantics of stable models, we use interpretations similar to those of the *here-and-there* intuitionistic logic (Heyting 1930; Pearce 1996). The main reasoning problem we consider is the existence of a stable model for a program and a dataset. We show in Section 4 that, in this setting, reasoning over the rational timeline is undecidable; furthermore, undecidability holds even for propositional forward-propagating programs (where rules cannot propagate information towards past time points) and to data containing only bounded intervals (i.e., where endpoints of all intervals are rational numbers). To regain decidability, in Section 5 we focus on the integer timeline. We show in Section 5.1 that discreteness of the timeline has a crucial influence on the computational behaviour, as reasoning becomes decidable and in EXPSPACE in data complexity; this is shown by exploiting Büchi automata and their complements to find candidate stable models and verify their minimality. Then, in Section 5.2 we show that, when restricted to forward-propagating (or, dually, to backwards-propagating) programs and bounded datasets, reasoning becomes PSPACE-complete and hence no harder than for negation-free DatalogMTL (Wałęga *et al.* 2019; 2020a). This is in stark contrast with the undecidability of the same fragment over the rational numbers.

2 Preliminaries

In this section we recapitulate the basics of temporal intervals over the integers or the rationals, and introduce the syntax and semantics of metric temporal operators.

2.1 *Timelines and temporal intervals*

A *timeline* $\mathbb{T}$ is either the set $\mathbb{Q}$ of rationals or the set $\mathbb{Z}$ of integers. A $\mathbb{T}$-*time point* is an element of $\mathbb{T}$. A $\mathbb{T}$-*interval* ϱ is a subset of $\mathbb{T}$ satisfying both of the following properties:

  – $t \in \varrho$ for all $t_1, t_2 \in \varrho$ and $t \in \mathbb{T}$ such that $t_1 < t < t_2$; and
  – the greatest lower bound ϱ^- and the least upper bound ϱ^+ of ϱ are both in $\mathbb{T} \cup \{-\infty, \infty\}$.

The bounds ϱ^- and ϱ^+ are called the *left* and *right endpoints* of ϱ, respectively. A $\mathbb{T}$-interval is *punctual* if it contains exactly one $\mathbb{T}$-time point, it is *non-negative* if it

Table 1. *Semantics of ground metric atoms*

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

contains no negative $\mathbb{T}$-time points, it is *bounded* if both its endpoints are in $\mathbb{T}$, and it is *closed* if it includes both of its endpoints. In these and similar notions, we often omit the reference to $\mathbb{T}$ if it is clear from the context. We consider binary representations of integers and fractional representations of rationals, with an integer numerator and a positive integer denominator, encoded in binary. We use standard representations of the form $\langle \varrho^-, \varrho^+ \rangle$ for a non-empty interval ϱ (i.e., $\varrho \cap \mathbb{T} \neq \emptyset$), where the *left bracket* $\langle$ is either $[$ or $($, the *right bracket* $\rangle$ is either $]$ or $)$, and, if numeric, the endpoints ϱ^- and ϱ^+ are represented as explained above. Brackets $[$ and $]$ indicate that the endpoints are included in the interval, whereas $($ and $)$ indicate that they are not included; note that, by this convention, $[$ and $]$ cannot be used with endpoints $-\infty$ and ∞. We often abbreviate a punctual interval $[t, t]$ as t. If it is clear from the context, we abuse notation and identify each interval representation with the interval it represents.

2.2 *Syntax and semantics of metric temporal expressions*

Assume a function-free first-order vocabulary and a timeline $\mathbb{T}$. A *relational atom* is an expression of the form $P(\mathbf{s})$, where P is a predicate and $\mathbf{s}$ is a tuple of constants and variables of the same arity as P. A *metric atom* is an expression given by the following grammar, where $P(\mathbf{s})$ ranges over relational atoms and ϱ over non-empty, non-negative intervals:

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

A metric atom is *ground* if it mentions no variables. A *metric fact* is an expression $M @ \varrho$, with M a ground metric atom and ϱ a non-empty $\mathbb{T}$-interval; it is *relational* if so is M. A *dataset* is a finite set of relational facts; it is *bounded* if so are all intervals it mentions. For a dataset $\mathscr{D}$, we denote with $t_{\mathscr{D}}^{\min}$ and $t_{\mathscr{D}}^{\max}$ the smallest and the largest numbers mentioned in $\mathscr{D}$; if $\mathscr{D}$ mentions no numbers, we let $t_{\mathscr{D}}^{\min} = t_{\mathscr{D}}^{\max} = 0$.

An *interpretation* $\mathfrak{I}$ is a function which assigns to each time point $t \in \mathbb{T}$ a set of ground relational atoms; if an atom $P(\mathbf{c})$ belongs to this set, we say that $P(\mathbf{c})$ is *satisfied* at t in $\mathfrak{I}$ and we write $\mathfrak{I}, t \models_{\mathbb{T}} P(\mathbf{c})$. This notion extends to other ground metric atoms as given in Table 1. Interpretation $\mathfrak{I}$ is a *model* of a metric fact $M @ \varrho$, written $\mathfrak{I} \models_{\mathbb{T}} M @ \varrho$, if $\mathfrak{I}, t \models M$ for all $t \in \varrho$; it is a *model* of a set $\mathscr{M}$ of metric facts (e.g., a dataset) if it is a model of all facts in $\mathscr{M}$. Set $\mathscr{M}$ *entails* a set $\mathscr{M}'$ of metric facts, written $\mathscr{M} \models \mathscr{M}'$, if every model of $\mathscr{M}$ is a model of $\mathscr{M}'$. An interpretation $\mathfrak{I}$ *contains* an interpretation $\mathfrak{I}'$, written $\mathfrak{I}' \subseteq \mathfrak{I}$, if for each ground relational atom $P(\mathbf{c})$ and each time point $t \in \mathbb{T}$, having $\mathfrak{I}', t \models_{\mathbb{T}} P(\mathbf{c})$ implies $\mathfrak{I}, t \models_{\mathbb{T}} P(\mathbf{c})$. Furthermore, $\mathfrak{I}$ is the *least* interpretation in a set X of interpretations, if $\mathfrak{I} \subseteq \mathfrak{I}'$ for every $\mathfrak{I}' \in X$.

3 DatalogMTL with negation under stable model semantics

In this section we propose DatalogMTL¬, which extends DatalogMTL with stratified negation as defined by Tena Cucala *et al.* (2021), to support unstratified use of negation in rules interpreted under stable model semantics.

The syntax of DatalogMTL¬ is the natural extension of the positive case: rule bodies are conjunctions of atoms and negated atoms, whereas rule heads are defined as in DatalogMTL. Forward-propagating DatalogMTL¬ programs are also defined analogously to the positive case (Wałęga *et al.* 2019), by requiring that rule bodies and heads do not mention metric operators referring to the future and to the past, respectively.

Definition 3.1
A *rule r* is an expression of the form

$$M \leftarrow M_1 \wedge \cdots \wedge M_k \wedge \text{not } M_{k+1} \wedge \cdots \wedge \text{not } M_m, \qquad (m \geq k \geq 0) \tag{1}$$

where each M_i is a metric atom, and M is a metric atom specified by the following grammar, where $P(\mathbf{s})$ ranges over relational atoms and ϱ over non-empty non-negative intervals:

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

The *head* of r is the consequent M and the *body* is the conjunction in the antecedent, where $M_1, \ldots, M_k$ are its *positive* body atoms, and $M_{k+1}, \ldots, M_m$ are its *negated* body atoms. A rule is *safe* if each variable it mentions in the head occurs in some positive body atom in a position other than a left operand of $\mathscr{S}$ or $\mathscr{U}$. A rule is *ground* if it has no variables, and it is *positive* if it has no negated body atoms. A (DatalogMTL¬) *program* is a finite set of safe rules; it is *ground* or *positive* if all its rules are. For a program Π, we let $\mathsf{ground}(\Pi)$ be the set of all ground rules that can be obtained by replacing variables in Π with constants. A program is *forward-propagating* (DatalogMTL¬_FP) if it is DatalogMTL¬ but does not mention the operators $\oplus$, $\boxplus$, and $\mathscr{U}$ in rule bodies, or the operator $\boxminus$ in rule heads.

The definition of stable models for Datalog with negation relies on the reduct construction by Gelfond and Lifschitz (1988), which has been adapted to various extensions of ASP (Faber *et al.* 2004). Such reduct constructions, however, do not have a natural equivalent in DatalogMTL¬, where interpretations may satisfy a fact at some, but not all points of the infinite timeline, and it is thus unclear which rules or atoms should be included in the reduct.

Following the approach of Cabalar and Vega (2007) and Cabalar *et al.* (2020), we define stable models for DatalogMTL¬ analogously to the models of *equilibrium logic* (Pearce 1996), which in turn are defined in terms of interpretations for the *here-and-there* intuitionistic logic (Heyting 1930). In this logic, each interpretation is an ordered pair (H, T) of sets H ("here") and T ("there") of relational propositional (i.e. using only predicates of arity 0) atoms such that $H \subseteq T$. We therefore start by generalising such interpretations to the context of DatalogMTL¬. For the remainder of this section, we fix a timeline $\mathbb{T}$, which will be implicit in all our definitions and technical results.

Definition 3.2
An *HT-interpretation* is a pair $(\mathfrak{H}, \mathfrak{T})$ of interpretations such that $\mathfrak{H} \subseteq \mathfrak{T}$. An HT-interpretation $(\mathfrak{H}, \mathfrak{T})$ is an *HT-model* of a dataset $\mathscr{D}$ if $\mathfrak{H}$ is a model of $\mathscr{D}$; furthermore, it

is an *HT-model* of a rule r if, for each rule of Form (1) in $\mathsf{ground}(\Pi)$ and for each $t \in \mathbb{T}$, the following hold:

1. If $\mathfrak{H}, t \models_\mathbb{T} M_i$ for all $i \in \{1, \ldots, k\}$ and $\mathfrak{T}, t \not\models_\mathbb{T} M_j$ for all $j \in \{k+1, \ldots, m\}$, then $\mathfrak{H}, t \models_\mathbb{T} M$.
2. If $\mathfrak{T}, t \models_\mathbb{T} M_i$ for all $i \in \{1, \ldots, k\}$ and $\mathfrak{T}, t \not\models_\mathbb{T} M_j$ for all $j \in \{k+1, \ldots, m\}$, then $\mathfrak{T}, t \models_\mathbb{T} M$.

Finally, $(\mathfrak{H}, \mathfrak{T})$ is an *HT-model* of a program if it is an HT-model of all its rules.

An HT-interpretation is, therefore, a pair $(\mathfrak{H}, \mathfrak{T})$ of standard interpretations. Interpretation $\mathfrak{H}$ is contained in $\mathfrak{T}$ and determines whether a dataset is satisfied. Although both interpretations are used to evaluate rules, it is $\mathfrak{T}$ which evaluates negated body atoms and thus determines when a rule with negation can be "triggered". When this happens, the rule ensures that if any of $\mathfrak{H}$ or $\mathfrak{T}$ satisfies the positive body atoms, then it will also satisfy the head. Since $\mathfrak{H} \subseteq \mathfrak{T}$ in an HT-interpretation, all relational facts entailed by $\mathfrak{H}$ are also entailed by $\mathfrak{T}$. We show next that this property can be generalised to arbitrary metric facts.

Proposition 3.3
For every HT-interpretation $(\mathfrak{H}, \mathfrak{T})$, metric atom M, and time point t, if $\mathfrak{H}, t \models M$ then $\mathfrak{T}, t \models M$.

Proof
We proceed by induction on the structure of M. If M is $\top$ or $\bot$, the claim holds trivially, and if M is a relational atom, then the claim holds by $\mathfrak{H} \subseteq \mathfrak{T}$. For the inductive step it suffices to consider the cases when M is of the form $\boxminus_\varrho M_1$ or $M_1 \mathscr{S}_\varrho M_2$, for some interval ϱ and metric atoms M_1 and M_2. Indeed, if M is of the form $\diamondsuit_\varrho M_1$ or $\Diamond_\varrho M_1$, then it is equivalent to $\top \mathscr{S}_\varrho M_1$ or $\top \mathscr{U}_\varrho M_1$, respectively, while the cases when M is of the form $\boxplus_\varrho M_1$ or $M_1 \mathscr{U}_\varrho M_2$ are symmetric to the cases of $\boxminus_\varrho M_1$ or $M_1 \mathscr{S}_\varrho M_2$, respectively.

If M is $\boxminus_\varrho M_1$, then $\mathfrak{H}, t \models M$ implies that $\mathfrak{H}, t' \models M_1$, for all t' such that $t - t' \in \varrho$. Hence, by the inductive hypothesis, $\mathfrak{T}, t' \models M_1$ for all t' such that $t - t' \in \varrho$, and so, $\mathfrak{T}, t \models \boxminus_\varrho M_1$. Similarly, if M is $M_1 \mathscr{S}_\varrho M_2$, then there is t' with $t - t' \in \varrho$ such that $\mathfrak{H}, t' \models M_2$ and $\mathfrak{H}, t'' \models M_1$, for all $t'' \in (t', t)$. By the inductive hypothesis we obtain that $\mathfrak{T}, t' \models M_2$ and $\mathfrak{T}, t'' \models M_1$, for all $t'' \in (t', t)$, so $\mathfrak{T}, t \models M_1 \mathscr{S}_\varrho M_2$. $\qquad\square$

Although the converse statement does not always hold, we can nonetheless prove the following result, which will underpin our definition of stable models.

Theorem 3.4
Let $(\mathfrak{T}, \mathfrak{T})$ be an HT-model of a program Π and a dataset $\mathscr{D}$. Then the set of interpretations $\{\mathfrak{H} \mid (\mathfrak{H}, \mathfrak{T}) \text{ is an HT-model of } \Pi \text{ and } \mathscr{D}\}$ contains a unique least interpretation.

Proof
We use transfinite induction to construct a sequence of interpretations $\mathfrak{H}_0, \mathfrak{H}_1, \ldots$, where each interpretation is contained in $\mathfrak{T}$. We will then show that $\mathfrak{H}_{\omega_1}$, where ω_1 is the first uncountable ordinal, is the least amongst all interpretations $\mathfrak{H}$ such that $(\mathfrak{H}, \mathfrak{T})$ is an HT-model of Π and $\mathscr{D}$.

Let $\mathfrak{H}_0$ be the least model of $\mathscr{D}$. Then, for each successor ordinal α, let $\mathfrak{H}_\alpha$ be the least interpretation satisfying the following property: for each rule of Form (1) in $\mathsf{ground}(\Pi)$,

and for each time point t, if $\mathfrak{H}_{\alpha-1}, t \models M_i$ for each $1 \leq i \leq k$ and $\mathfrak{T}, t \not\models M_j$ for each $k + 1 \leq j \leq m$, then $\mathfrak{H}_\alpha, t \models M$. Moreover, for each limit ordinal α, we define $\mathfrak{H}_\alpha$ as $\bigcup_{\beta < \alpha} \mathfrak{H}_\beta$. By induction on ordinals α we can show simultaneously that $\mathfrak{H}_\alpha$ is well-defined and that $\mathfrak{H}_\alpha \subseteq \mathfrak{T}$. For the basis of the induction, we recall that $\mathfrak{H}_0$ is the least model of $\mathscr{D}$, so $\mathfrak{H}_0$ is well-defined. Moreover, since $(\mathfrak{T}, \mathfrak{T})$ is an HT-model of $\mathscr{D}$, we obtain that $\mathfrak{H}_0 \subseteq \mathfrak{T}$. Now, consider the inductive step for a successor ordinal α. To show that $\mathfrak{H}_\alpha$ is well-defined it suffices to show that for each rule of Form (1) in $\mathsf{ground}(\Pi)$, and for each time point t, if $\mathfrak{H}_{\alpha-1}, t \models M_i$ for each $1 \leq i \leq k$ and $\mathfrak{T}, t \not\models M_j$ for each $k + 1 \leq j \leq m$, then M is not $\bot$. Indeed, by the inductive assumption we have $\mathfrak{H}_{\alpha-1} \subseteq \mathfrak{T}$, so if M is $\bot$, then $\mathfrak{T}, t \models \bot$, which contradicts the assumption that $(\mathfrak{T}, \mathfrak{T})$ is an HT-model of Π. Moreover, since $\mathfrak{H}_{\alpha-1} \subseteq \mathfrak{T}$ and $(\mathfrak{T}, \mathfrak{T})$ is an HT-model of Π, we need to have $\mathfrak{T}, t \models M$, so $\mathfrak{H}_\alpha \subseteq \mathfrak{T}$. The inductive step for a limit ordinal α holds trivially, since $\mathfrak{H}_\alpha$ is defined as $\bigcup_{\beta < \alpha} \mathfrak{H}_\beta$.

We thus obtain that $\mathfrak{H}_{\omega_1} \subseteq \mathfrak{T}$, and so, $(\mathfrak{H}_{\omega_1}, \mathfrak{T})$ is an HT-interpretation. By construction, $\mathfrak{H}_{\omega_1}$ contains $\mathfrak{H}_0$ and therefore $(\mathfrak{H}_{\omega_1}, \mathfrak{T})$ is an HT-model of $\mathscr{D}$. It is also an HT-model of Π, since ω_1 rounds of rule applications are enough to ensure that $\mathfrak{H}_{\omega_1}$ is a fixpoint with respect to the application of the rules of Π (Brandt *et al.* 2017). Finally, to show that $\mathfrak{H}_{\omega_1}$ is the least among interpretations $\mathfrak{H}$ such that $(\mathfrak{H}, \mathfrak{T})$ is an HT-model of Π and $\mathscr{D}$, consider any such $\mathfrak{H}$. Using transfinite induction in a way similar to the previous paragraph, one can show that $\mathfrak{H}_\alpha \subseteq \mathfrak{H}$ for each ordinal α, and thus $\mathfrak{H}_{\omega_1} \subseteq \mathfrak{H}$. $\square$

Given a program Π, a dataset $\mathscr{D}$, and an interpretation $\mathfrak{T}$ such that $(\mathfrak{T}, \mathfrak{T})$ is an HT-model of Π, we let $\mathfrak{H}_{\Pi, \mathscr{D}}^{\mathfrak{T}}$ denote the least interpretation whose existence is guaranteed by Theorem 3.4.

In equilibrium logic, a model of a program is a set T of relational propositional atoms that satisfies the rules of the program and for which there exists no set $H \subsetneq T$ such that (H, T) is a model of the program in here-and-there logic. This ensures that equilibrium logic implements a kind of minimal model reasoning. We next generalise this notion to DatalogMTL$^\lnot$ by building on our previous definition of the least interpretation $\mathfrak{H}_{\Pi, \mathscr{D}}^{\mathfrak{T}}$.

Definition 3.5
An interpretation $\mathfrak{T}$ is a *stable model* of a program Π and a dataset $\mathscr{D}$ if and only if $(\mathfrak{T}, \mathfrak{T})$ is an HT-model of Π and $\mathscr{D}$, and $\mathfrak{H}_{\Pi, \mathscr{D}}^{\mathfrak{T}} = \mathfrak{T}$.

Example 3.6
Consider a dataset with a single fact $P@[0, 1]$ and a propositional DatalogMTL$^\lnot$ program consisting of a single rule $R \leftarrow \diamondminus_1 P \wedge \mathsf{not}\, Q$. In this setting, there is just a single stable model, namely the interpretation where P holds at all time points from $[0, 1]$, R holds at all time points from $[1, 2]$, and no relational atoms are satisfied anywhere else.

Next, consider a dataset with facts $P@0$ and $Q@1$, together with a propositional DatalogMTL$^\lnot$ program that consists of two rules $R \leftarrow P \wedge \mathsf{not}\, \diamondplus_1 R$ and $R \leftarrow Q \wedge \mathsf{not}\, \diamondminus_1 R$. This dataset and program have two stable models. In the first model, P and R hold at 0, whereas Q holds at 1. In the second model, P holds at 0, whereas Q and R hold at 1.

Finally, let us consider the empty dataset and a program consisting of rules $Q \leftarrow \mathsf{not}\, P$ and $P \leftarrow \mathsf{not}\, Q$. Syntactically, this is not only apropositional DatalogMTL$^\lnot$ program, but

also a standard ASP. According to our temporal semantics, this program and dataset admit infinitely many stable models: for each set X of time points, there is a stable model in which P holds at each time point in X and Q holds at all other time points. In contrast, the same program under the standard ASP semantics has only two stable models, namely $\{P\}$ and $\{Q\}$.

We next show that our semantics for DatalogMTL$^\neg$ also generalises the semantics of (positive) DatalogMTL programs. If a DatalogMTL program Π and a dataset $\mathscr{D}$ have a model, they also admit a least model (Brandt *et al.* 2017). This can be equivalently reformulated by stating that if the set of all interpretations $\{\mathfrak{I} \mid (\mathfrak{I}, \mathfrak{I})$ is an HT-model of Π and $\mathscr{D}\}$ is not empty, then this set contains a unique least interpretation.

Theorem 3.7
Let Π be a positive program and let $\mathscr{D}$ be a dataset. An interpretation $\mathfrak{I}$ is a stable model of Π and $\mathscr{D}$ if and only if $\mathfrak{I}$ is their least model.

Proof
We can first use the fact that Π is positive to show that if $(\mathfrak{I}, \mathfrak{I})$ is an HT-model of Π and $\mathscr{D}$, then $\mathfrak{H}^{\mathfrak{I}}_{\Pi,\mathscr{D}}$ is the least model of Π and $\mathscr{D}$. Indeed, if $(\mathfrak{I}, \mathfrak{I})$ is an HT-model of Π and $\mathscr{D}$, we can define the sequence $\mathfrak{H}_0, \mathfrak{H}_1, \ldots$ of interpretations contained in $\mathfrak{I}$ as in the proof of Theorem 3.4, which satisfies $\mathfrak{H}_{\omega_1} = \mathfrak{H}^{\mathfrak{I}}_{\Pi,\mathscr{D}}$. Furthermore, since Π is positive, we can observe that, for every ordinal α, it holds that $\mathfrak{H}_\alpha = T^\alpha_\Pi(\mathfrak{I}_{\mathscr{D}})$, where $T^\alpha_\Pi(\mathfrak{I}_{\mathscr{D}})$ is the result of applying α times the immediate consequence operator of a positive program Π to an interpretation $\mathfrak{I}_{\mathscr{D}}$ represented by $\mathscr{D}$. In particular, $\mathfrak{H}_{\omega_1} = T^{\omega_1}_\Pi(\mathfrak{I}_{\mathscr{D}})$, which is the least model of Π and $\mathscr{D}$ (Wałęga *et al.* 2021; Brandt *et al.* 2017). Hence, $\mathfrak{H}^{\mathfrak{I}}_{\Pi,\mathscr{D}}$ is the least model of Π and $\mathscr{D}$.

Now, if $\mathfrak{I}$ is a stable model of Π and $\mathscr{D}$, then $(\mathfrak{I}, \mathfrak{I})$ is an HT-model of Π and $\mathscr{D}$; as shown in the previous paragraph, this implies that $\mathfrak{H}^{\mathfrak{I}}_{\Pi,\mathscr{D}}$ is the least model of Π and $\mathscr{D}$. However, since $\mathfrak{I}$ is a stable model of Π and $\mathscr{D}$, we have $\mathfrak{I} = \mathfrak{H}^{\mathfrak{I}}_{\Pi,\mathscr{D}}$, and thus $\mathfrak{I}$ is also the least model of Π and $\mathscr{D}$. Conversely, if $\mathfrak{I}$ is the least model of Π and $\mathscr{D}$, then $(\mathfrak{I}, \mathfrak{I})$ is an HT-model of Π and $\mathscr{D}$; then, as shown in the previous paragraph, $\mathfrak{I} = \mathfrak{H}^{\mathfrak{I}}_{\Pi,\mathscr{D}}$, and so $\mathfrak{I}$ is a stable model of Π and $\mathscr{D}$. □

It follows that, if a positive program and a dataset have a model, then they have a stable model. Note, however, that this is not the case for other temporal logics with stable model semantics (Cabalar and Demri 2011; Bozzelli and Pearce 2015), and the reason why this property holds in our setting is given by Theorem 3.4.

Finally, our semantics also generalises that of stratifiable DatalogMTL$^\neg$ programs (Tena Cucala *et al.* 2021), where rules do not have cyclic dependencies via negation and can be organised in strata. Each such a stratifiable, $\perp$-free program Π and dataset $\mathscr{D}$ have a single least model constructed by applying to $\mathscr{D}$ rules of Π stratum by stratum in a minimal way (Tena Cucala *et al.* 2021). As in the case of positive programs, we can show that such least model corresponds to the single stable model of Π and $\mathscr{D}$. Hence, analogously to the case of plain Datalog, positive and stratifiable DatalogMTL programs cannot have multiple stable models. Arbitrary programs, however, can have any number of stable models, which is witnessed by Example 3.6.

In the rest of the paper we study decidability and complexity of *reasoning*, which is (in the context of this paper) the problem of checking if a DatalogMTL$^\neg$ program Π and a dataset $\mathscr{D}$ have a stable model. We focus on *data complexity* – that is, we assume that Π is fixed and only $\mathscr{D}$ forms the input – which is the most relevant measure if complexity in data intensive applications.

Before we close this section, however, it is worth pointing out the connections between the problem of checking existence of a stable model and the related problem of *fact entailment*, as defined next. Following the standard conventions of non-monotonic logics and ASP (Eiter *et al.* 2009), we say that a DatalogMTL$^\neg$ program Π and a dataset $\mathscr{D}$ *bravely* entail a relational fact $P(\mathbf{c})@\varrho$ if $\mathfrak{I} \models_{\mathbb{T}} P(\mathbf{c})@\varrho$ for some stable model $\mathfrak{I}$ of Π and $\mathscr{D}$, and we say that Π and $\mathscr{D}$ *cautiously* entail $P(\mathbf{c})@\varrho$ if $\mathfrak{I} \models_{\mathbb{T}} P(\mathbf{c})@\varrho$ for all stable models $\mathfrak{I}$ of Π and $\mathscr{D}$. The problem of brave (resp. cautious) fact entailment consists in deciding whether a DatalogMTL$^\neg$ program and a dataset bravely (resp. cautiously) entail a given relational fact. As we show next, both variants of the problem are inter-reducible with checking existence (or non-existence) of a stable model. Moreover, we argue that these reductions allow us to transfer bounds for *data complexity* which, for fact entailment, refers to the setting where both the program Π and the fact $P(\mathbf{c})@\varrho$ are fixed, and only the dataset $\mathscr{D}$ constitutes the input.

Proposition 3.8
In DatalogMTL$^\neg$, existence of a stable model can be reduced in AC^0 to (i) brave fact entailment, and to (ii) the complement of cautious fact entailment, and vice versa. Furthermore, the reductions involved do not depend on the input dataset.

Proof
We start by showing Statement (i). To check if a DatalogMTL$^\neg$ program Π and a dataset $\mathscr{D}$ have a stable model, it suffices to add to $\mathscr{D}$ a fact $P@0$ with a fresh proposition P and check whether Π and the extended dataset bravely entail $P@0$. To check if Π and $\mathscr{D}$ bravely entail a relation fact $P(\mathbf{c})@\varrho$, it suffices to verify that the following program Π' and dataset $\mathscr{D}'$ have a stable model:

$$\Pi' = \Pi \cup \{\bot \leftarrow P'(\mathbf{x}) \wedge \mathsf{not}\, \boxminus_{\varrho_1} P(\mathbf{x}), \bot \leftarrow P'(\mathbf{x}) \wedge \mathsf{not}\, \boxplus_{\varrho_2} P(\mathbf{x})\}, \quad \mathscr{D}' = \mathscr{D} \cup \{P'(\mathbf{c})@t\},$$

where P' is a fresh predicate of the same arity as P, $\mathbf{x}$ is a tuple of distinct variables, t is an arbitrary time point belonging to ϱ, whereas ϱ_1 and ϱ_2 depend on both ϱ and t; for example, if $\varrho = [t_1, t_2]$, then $\varrho_1 = [0, t - t_1]$ and $\varrho_2 = [0, t_2 - t)$, where if $t_2 = \infty$, then $t_2 - t$ stands for ∞.

Next, we show Statement (ii). To check if Π and $\mathscr{D}$ have a stable model, it suffices to check if they do not cautiously entail a fact $P@0$, where P is a fresh proposition (Brandt *et al.* 2018, Proposition 3). On the other hand, to check if Π and $\mathscr{D}$ do not cautiously entail a relational fact $P(\mathbf{c})@\varrho$, it suffices to verify that the following program Π'' and dataset $\mathscr{D}''$ have a stable model:

$$\Pi'' = \Pi \cup \{\bot \leftarrow P'(\mathbf{x}) \wedge \boxminus_{\varrho_1} P(\mathbf{x}) \wedge \boxplus_{\varrho_2} P(\mathbf{x})\}, \qquad \mathscr{D}'' = \mathscr{D} \cup \{P'(\mathbf{c})@t\},$$

where P', $\mathbf{x}$, t, ϱ_1, and ϱ_2 are as in the proof of Statement (i).

Finally, we observe that all the above reductions can be performed in AC^0. Moreover, they allow us to transfer data complexity bounds, since the programs and facts we construct in the reductions do not depend on input datasets. $\qquad\square$

4 Undecidability over the rational timeline

In this section we focus on the rational timeline, so let us fix $\mathbb{T} = \mathbb{Q}$. In this setting, standard reasoning problems are PSpace-complete in data complexity for positive programs (Wałęga *et al.* 2019), as well as for programs with negation, if they are stratified (Tena Cucala *et al.* 2021).

We next show that, in stark contrast with the positive case, reasoning in DatalogMTL$^{\neg}$ is undecidable. Furthermore, undecidability holds even for programs that are propositional, forward-propagating and considered to be fixed, and even if the input datasets are bounded.

Theorem 4.1
Checking whether a propositional DatalogMTL$^{\neg}_{\mathsf{FP}}$ and a bounded dataset have a stable model over the rational timeline is undecidable with respect to data complexity.

Proof
Let $\mathfrak{M} = (\Sigma, \mathcal{Q}, \delta, q_{init}, q_{halt})$ be a deterministic Turing machine with Σ the input alphabet, $\mathcal{Q}$ the set of states, $\delta : \Sigma_{\sqcup} \times (\mathcal{Q} \setminus \{q_{halt}\}) \longrightarrow \Sigma_{\sqcup} \times \mathcal{Q} \times \{\mathsf{L}, \mathsf{R}\}$ the transition function for $\Sigma_{\sqcup} = \Sigma \cup \{\sqcup\}$ and blank symbol $\sqcup$, L and R the symbols indicating the head movements, and $q_{init}, q_{halt} \in \mathcal{Q}$ the initial and halting states. Without loss of generality, we assume that $\mathfrak{M}$ never tries to move to the left of the left-most cell of the tape.

We construct a propositional DatalogMTL$^{\neg}_{\mathsf{FP}}$ program $\Pi_{\mathfrak{M}}$ and then reduce (in AC^0) every input word w to a dataset $\mathcal{D}_w$ with only bounded intervals so that $\mathfrak{M}$ halts on w if and only if $\Pi_{\mathfrak{M}}$ and $\mathcal{D}_w$ do not have a stable model.

We represent, for each $i \geq 1$, the ith configuration in the computation of $\mathfrak{M}$ on input w within the interval $[2i, 2i+2)$, as illustrated in Figure 1, where we assume that in the configuration the state is q, the head is over the jth cell, and the contents of the first $|w| + i$ cells of the tape are symbols $s_1, \ldots, s_{|w|+i}$ (in the ith configuration, only the first $|w| + i$ cells can be non-empty). The state is encoded within the first half $[2i, 2i+1]$ of the interval: a proposition S_q holds at some time point within $[2i, 2i+1]$. The contents of the tape and the head position are encoded within the second half $(2i+1, 2i+2)$ of the interval; in particular, $|w| + i$ time points $t^i_1 < \cdots < t^i_{|w|+i}$ in $(2i+1, 2i+2)$ are used so that, for each $j \in \{1, \ldots, |w| + i\}$, proposition C_{s_j} holds at t^i_j, encoding the fact that s_j are the contents of the jth cell in the configuration, and proposition H holds at t^i_j, encoding the fact that the head is over the jth cell in the configuration. We also use additional propositions: S, which holds all through $[2i, 2i+1]$ and ensures that these intervals are only used to encode states; N, which holds at a single new time point in $(2i+1, 2i+2)$ beyond $t^i_{|w|+i}$ and will allow us to increase the number of time points encoding cells; $\overline{N}$ and $\overline{H}$, which simulate negations of N and H, respectively; $\overline{C}$, which
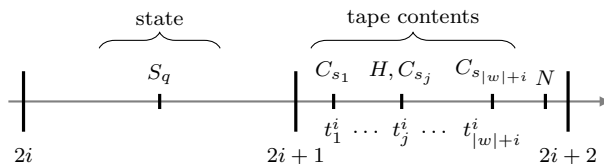

Fig. 1. Encoding of the ith configuration.

marks points not used to encode the tape contents, and L, used for encoding left-moving transitions.

The first block of rules in $\Pi_{\mathfrak{M}}$ consists of the following rules, for each $X \in \{N, H\}$ and $s \in \Sigma_{\sqcup}$:

$$\overline{X} \leftarrow \mathsf{not}\, X, \qquad X \leftarrow \mathsf{not}\, \overline{X}, \qquad \bot \leftarrow X \wedge \overline{X}, \qquad \bot \leftarrow X \wedge \Diamondplus_{(0,1)} X,$$

$$\overline{X} \leftarrow S, \qquad \overline{N} \leftarrow C_s, \qquad \overline{H} \leftarrow \overline{C}, \qquad \bot \leftarrow S \wedge (\overline{C} \wedge \overline{N}) \mathscr{S}_{(0,1)} C_s.$$

The first three rules state that, at each time point, either X or $\overline{X}$ holds. The fourth rule states that X cannot hold twice in an open interval of length 1. By the fifth rule, X and S cannot hold at the same time point. The sixth rule states that $\overline{N}$ holds in all time points encoding cells. The second to last rule states that H does not hold in time points that do not encode cells. The last rule ensures that after time point $t^i_{|w|+i}$ encoding the last relevant cell in the ith configuration, there is another time point within $(2i+1, 2i+2)$ where N holds. Note that the last rule uses conjunction within a metric operator, which is not syntactically allowed, but can be easily simulated by replacing $\overline{C} \wedge \overline{N}$ with a fresh proposition P and adding a rule $P \leftarrow \overline{C} \wedge \overline{N}$; this abbreviation will be useful for simplifying other formulas used later on in the reduction.

The second block consists of the following rules, propagating propositions to the interval encoding the consequent configuration, for every $s \in \Sigma_{\sqcup}$:

$$\boxplus_2 S \leftarrow S, \qquad \boxplus_2 \overline{C} \leftarrow \overline{C} \wedge \overline{N}, \qquad \boxplus_2 C_{\sqcup} \leftarrow N \wedge \Diamondplus_{(0,\infty)} S_{q_{init}}, \qquad \boxplus_2 C_s \leftarrow C_s \wedge \overline{H}.$$

By the first rule, S is always propagated into the future from t to $t+2$. The second rule states that, if t does not encode a cell and $\overline{N}$ holds therein, then $t+2$ does not encode a cell either. By the third rule, if N holds at t and this t is to the right of the time point encoding the initial state, then $t+2$ encodes an empty cell. The last rule states that, if t encodes a cell with contents s and the head is not on this cell, then $t+2$ also encodes a cell with contents s.

We next encode the left-moving transitions. Proposition L is used to indicate a time point encoding a cell such that the head was on it in the previous configuration and then moved to the left. Program $\Pi_{\mathfrak{M}}$ contains the following rules for every $s \in \Sigma$ and $q \in \mathscr{Q}$ with transition $\delta(s, q) = (s', q', \mathsf{L})$, and every $s^* \in \Sigma$:

$$\boxplus_2 L \wedge \boxplus_1 S_{q'} \wedge \boxplus_2 C_{s'} \leftarrow H \wedge C_s \wedge \Diamondplus_{(0,2)} S_q,$$

$$\bot \leftarrow L \wedge \boxminus_{(0,1)} \overline{H},$$

$$\bot \leftarrow L \wedge \Diamondplus_{(0,1)} (C_{s^*} \wedge \Diamondplus_{(0,1)} H).$$

The first rule simulates the transition: H holds as intended, the state is changed from q to q', and the contents of the cell under the head change from s to s' (the conjunction in the head is used for brevity and can be simulated by several rules). The last two rules encode the position of the head in the consequent configuration, by stating that H holds at the first time point encoding a cell to the left of the time point with L.

Similarly, for each transition $\delta(s, q) = (s', q', \mathsf{R})$ moving the head to the right and any $s^* \in \Sigma$, program $\Pi_{\mathfrak{M}}$ has the rules

$$\boxplus_1 S_{q'} \wedge \boxplus_2 C_{s'} \leftarrow H \wedge C_s \wedge \Diamondplus_{(0,2)} S_q,$$

$$\boxplus_2 H \leftarrow C_{s^*} \wedge \overline{C} \mathscr{S}_{(0,1)} (H \wedge C_s) \wedge \Diamondplus_{(0,2)} S_q,$$

$$\boxplus_2 H \leftarrow N \wedge \overline{C} \mathscr{S}_{(0,1)} (H \wedge C_s) \wedge \Diamondplus_{(0,2)} S_q.$$

Here, the first rule encodes the change of the state and the contents of the cell above which the head is. The last two rules simulate the change of the position of the head. Finally, $\Pi_\mathfrak{M}$ contains rule $\bot \leftarrow S_{q_{halt}}$, which yields an inconsistency when the halting state is reached.

We next reduce an input word $w = s_1 \ldots s_{|w|}$ to a dataset $\mathscr{D}_w$. Assuming that w is non-empty, we let $t_k = 1 + \frac{k}{|w|+1}$ for each $k \in \{1, \ldots, |w|\}$ (it is only important here that $1 < t_1 < \cdots < t_{|w|} < 2$); then, $\mathscr{D}_w$ contains the facts:

$$S@[0,1], \qquad \overline{N}@[0,1], \qquad \overline{H}@[0,1], \qquad S_{q_{init}}@1, \qquad H@t_1,$$
$$C_{s_1}@t_1, \ \ldots, \ C_{s_{|w|}}@t_{|w|}, \qquad \overline{C}@(1, t_1), \qquad \overline{C}@(t_1, t_2), \ \ldots, \ \overline{C}@(t_{|w|}, 2).$$

Intuitively, $\mathscr{D}_w$ describes the initial configuration of $\mathfrak{M}$ on w within $[0,2)$; the initial state is encoded in 1 and $t_1, \ldots, t_{|w|}$ encode the first $|w|$ cells of $\mathfrak{M}$. Moreover, $\overline{C}$ holds in all other time points in $(1,2)$, whereas $\overline{N}$ and $\overline{H}$ hold in $[0,1]$.

We next show that $\Pi_\mathfrak{M}$ and $\mathscr{D}_w$ have a stable model if and only if $\mathfrak{M}$ does not halt on w. Assume that $\mathfrak{T}$ is a stable model of $\Pi_\mathfrak{M}$ and $\mathscr{D}_w$. Then, using induction over $i \in \mathbb{N}$, we can prove that the ith configuration in the computation of $\mathfrak{M}$ on w is encoded as discussed above. In particular, if q is the state of $\mathfrak{M}$ in the ith configuration, then $\mathfrak{T}, t \models S_q$ for some $t \in [2i, 2i+1]$. Since $\mathfrak{T}$ is a model of $\Pi_\mathfrak{M}$, however, it satisfies the rule $\bot \leftarrow S_{q_{halt}}$; therefore, the state q_{halt} cannot occur in any configuration in the computation of $\mathfrak{M}$ on w, and so $\mathfrak{M}$ does not halt on w.

For the opposite direction assume that $\mathfrak{M}$ does not halt on w. Then, we let $\mathfrak{T}$ be the minimal interpretation which satisfies $\mathscr{D}_w$ and the following statements, for every positive integer i, where we let $t_j^i = 2i + 1 + \sum_{k=1}^j \frac{1}{2^k}$ for each $j \in \mathbb{N}$ (this definition ensures that there are infinitely many time points of the form t_j^i in interval $(2i+1, 2i+2)$):

– $\mathfrak{T} \models S@[2i, 2i+1]$,
– $\mathfrak{T}, t_j^i \models C_s$, whenever s are the contents of the jth cell of $\mathfrak{M}$ in the ith step of computation on w, for $j \in \{1, \ldots, |w| + i\}$,
– $\mathfrak{T}, t \models \overline{C}$, for each $t \in (2i, 2i+1) \setminus \{t_1^i, \ldots, t_{|w|+i}^i\}$,
– $\mathfrak{T}, t_{|w|+i+1}^i \models N$ and $\mathfrak{T}, t' \models \overline{N}$ for each $t' \in [2i, 2i+2) \setminus \{t_{|w|+i+1}^i\}$,
– $\mathfrak{T}, t_j^i \models H$ if the head of $\mathfrak{M}$ is above the jth cell in the ith step of its computation on w,
– $\mathfrak{T}, t \models \overline{H}$ for each $t \in [2i, 2i+2) \setminus \{t_j^i\}$, for j such that the head of $\mathfrak{M}$ is above the jth cell in the ith step of its computation on w,
– $\mathfrak{T}, t_j^{i+1} \models L$ if the head of $\mathfrak{M}$ is above the jth cell in the ith step of its computation on w,
– $\mathfrak{T}, t_j^i + 1 \models S_q$ if $\mathfrak{T}, t_j^i \models H$ and $\mathfrak{M}$ is in the state q in the $i+1$th step of computation on w.

Clearly, $(\mathfrak{T}, \mathfrak{T})$ is an HT-model of $\mathscr{D}_w$, and it can be also verified, by inspecting the rules in $\Pi_\mathfrak{M}$, that $(\mathfrak{T}, \mathfrak{T})$ is an HT-model of $\Pi_\mathfrak{M}$. Finally, to show that $(\mathfrak{T}, \mathfrak{T})$ is a stable model of $\Pi_\mathfrak{M}$ and $\mathscr{D}_w$, we need to show that $\mathfrak{T} = \mathfrak{H}_{\Pi_\mathfrak{M}, \mathscr{D}_w}^\mathfrak{T}$. Towards this goal, we first construct the sequence $\mathfrak{H}_0, \mathfrak{H}_1$ of interpretations as in the proof of Theorem 3.4, for which it holds that $\mathfrak{H}_{\omega_1} = \mathfrak{H}_{\Pi_\mathfrak{M}, \mathscr{D}_w}^\mathfrak{T}$. Then we can easily show, using transfinite induction, that for each $i \in \mathbb{N}$ and each relational fact $M@t$ with $t \in [2i, 2i+2)$, $\mathfrak{T} \models M@t$ if and only if $\mathfrak{H}_i \models M@t$, and also that $\mathfrak{H}_i \models M@t$ if and only if $\mathfrak{H}_{\Pi_\mathfrak{M}, \mathscr{D}_w}^\mathfrak{T} \models M@t$, which together imply $\mathfrak{T} = \mathfrak{H}_{\Pi_\mathfrak{M}, \mathscr{D}_w}^\mathfrak{T}$. $\qquad \square$

Observe that the reduction above shows undecidability of reasoning in DatalogMTL¬ over the rational timeline, even if we restrict our attention to fixed propositional programs in the forward-propagating fragment, and we consider only bounded datasets. In the next section, we turn our attention to the integer timeline and show that reasoning becomes decidable.

5 Decidability over the integer timeline

In this section we consider the integer timeline and thus we fix $\mathbb{T} = \mathbb{Z}$. We will show that, in this case, reasoning becomes decidable in ExpSpace with respect to data complexity; furthermore, complexity drops to PSpace if we restrict our attention to forward-propagating programs and datasets mentioning only bounded intervals – a setting well-suited for stream reasoning (Wałęga *et al.* 2019; Ronca *et al.* 2018). In this setting, the additional expressive power provided by stable models comes at no computational cost since reasoning in the corresponding positive fragment is already PSpace-complete (Wałęga *et al.* 2020a; 2019).

In prior work on positive and stratifiable programs, upper bounds for reasoning have been established by constructing generalised Büchi automata that accept (words describing) models of a given program and dataset (Wałęga *et al.* 2020a; Tena Cucala *et al.* 2021). Checking existence of a stable model is more demanding, as we additionally need to ensure model minimality as in Definition 3.5; this requirement is non-trivial, and we will handle it differently for the cases of arbitrary and forward-propagating programs.

In the general case (Section 5.1), we construct two kinds of left- and right-moving automata: the first kind allows us to check existence of an HT-model of the form $(\mathfrak{T}, \mathfrak{T})$, while the second kind allows us to check existence of an HT-model of the form $(\mathfrak{H}, \mathfrak{T})$ with $\mathfrak{H} \neq \mathfrak{T}$. Then, a pair of words $\overleftarrow{w}$ and $\overrightarrow{w}$ that are accepted, respectively, by a pair of left- and right-moving automata of the first kind, but not by any pair of left and right-moving automata of the second kind, represents a stable model. This construction is conceptually similar to that of Cabalar and Demri (2011) for a logic with linear temporal operators and involves complementing nondeterministic automata, which leads to an exponential blowup. Consequently, we obtain an ExpSpace upper bound and thus an exponential gap in data complexity with respect to positive programs (Wałęga *et al.* 2020a). In the case of forward-propagating programs (Section 5.2) we propose a different construction exploiting the fact that rules propagate information in a single temporal direction. This allows us to build automata that guarantee model minimality without complementation. As a result, we can establish a tight PSpace bound in data complexity.

5.1 General programs

It will be convenient for our presentation to assume that programs are in a normal form analogous to that by Tena Cucala *et al.* (2021) for stratifiable programs: in each normalised rule the head is a relational atom or $\bot$, there is neither nesting of metric operators nor occurrences of $\Leftrightarrow$ or $\oplus$ in rule bodies, and the only unbounded interval allowed is $[0, \infty)$.

Proposition 5.1

Each program Π can be normalised in polynomial time into a program Π' such that, for each dataset $\mathscr{D}$, program Π and dataset $\mathscr{D}$ have a stable model if and only if so do Π' and $\mathscr{D}$.

Proof

To construct Π' we first delete all (trivial) rules having $\top$ as the head. Then, we eliminate from the remaining rule heads metric operators (i.e., boxes). To this end, we replace each rule of the form $\Box^1_{\varrho_1} \ldots \Box^n_{\varrho_n} P(\mathbf{s}) \leftarrow B$, where $n \geq 0$, each $\Box^i$ is either $\boxminus$ or $\boxplus$, and B is the body of the rule, with rules $P'(\mathbf{s}) \leftarrow B$ and $P(\mathbf{s}) \leftarrow \Diamond^1_{\varrho_1} \cdots \Diamond^n_{\varrho_n} P'(\mathbf{s})$, where $\Diamond^i = \diamondplus$ if $\Box^i = \boxminus$ and otherwise $\Diamond^i = \diamondminus$, and P' is a fresh predicate of the same arity as P.

Second, we iteratively eliminate nested temporal operators from rule bodies. To this end, consider a rule r whose body has an occurrence M of a metric atom that mentions only one temporal operator and which is in the scope of some other temporal operator. If M is of one of the forms $\diamondminus_\varrho P(\mathbf{s})$, $\diamondplus_\varrho P(\mathbf{s})$, $\boxminus_\varrho P(\mathbf{s})$, or $\boxplus_\varrho P(\mathbf{s})$, then we replace it with $P'(\mathbf{s})$ and add a rule $P'(\mathbf{s}) \leftarrow M$, where P' is a fresh predicate of the same arity as P. If M mentions $\mathscr{S}$ or $\mathscr{U}$, we need to proceed in a special way to ensure safety of the new rules. If M is of the form $P_1(\mathbf{s}_1)\mathscr{S}_\varrho P_2(\mathbf{s}_2)$, we remove r and proceed as follows (note that the conditions below are not exclusive):

- if $0 \in \varrho$, we add the rule obtained by replacing M in r with $P_2(\mathbf{s}_2)$,
- if $1 \in \varrho$, we add the rule obtained by replacing M in r with $\boxminus_1 P_2(\mathbf{s}_2)$,
- we add the rule $P'(\mathbf{s}_1, \mathbf{s}_2) \leftarrow P_1(\mathbf{s}_1)\mathscr{S}_\varrho P_2(\mathbf{s}_2) \wedge \diamondminus_{[0,\infty)} P_1(\mathbf{s}_1)$ and the rule obtained by replacing M in r with $P'(\mathbf{s}_1, \mathbf{s}_2)$, where P' is a fresh predicate whose arity equals the sum of arities of P_1 and P_2.

If M is of the form $P_1(\mathbf{s}_1)\mathscr{U}_\varrho P_2(\mathbf{s}_2)$, we proceed analogously to the case when M is of the form $P_1(\mathbf{s}_1)\mathscr{S}_\varrho P_2(\mathbf{s}_2)$, but instead of $\boxminus$ and $\diamondminus$ we use, respectively, $\boxplus$ and $\diamondplus$ in the new rules. Moreover, if M mentions $\top$ or $\bot$, we treat these symbols as nullary predicates and proceed as before.

Next, we eliminate diamond operators by replacing $\diamondminus_\varrho M$ and $\diamondplus_\varrho M$ with, respectively, $\top \mathscr{S}_\varrho M$ and $\top \mathscr{U}_\varrho M$. Finally, we eliminate unbounded intervals ϱ different from $[0,\infty)$ as follows. If a rule r mentions $\Box_\varrho P(\mathbf{s})$, we replace this $\Box_\varrho P(\mathbf{s})$ with $\Box_t P(\mathbf{s})$ and add a rule $P'(\mathbf{s}) \leftarrow \Box_{[0,\infty)} P(\mathbf{s})$, where $\Box$ is either $\boxminus$ or $\boxplus$, t is the least natural number in ϱ (so $t \geq 1$), and P' is a fresh predicate of the same arity as P. In the case of operators $\mathscr{S}$ and $\mathscr{U}$ we need to pay special attention to ensure that the new rules are safe. Assume that a rule r mentions $M = P_1(\mathbf{s}_1)\mathscr{S}_\varrho P_2(\mathbf{s}_2)$ with an unbounded $\varrho \neq [0,\infty)$. We remove r and proceed as follows, for t the least natural number belonging to ϱ:

- if $1 \in \varrho$, we add the rule obtained by replacing M in r with $\boxminus_1 P_2(\mathbf{s}_2)$,
- if $1 \notin \varrho$, we add the rule obtained by replacing M in r with $\boxminus_t P_2(\mathbf{s}_2) \wedge \boxminus_{(0,t)} P_1(\mathbf{s}_1)$
- we add the rule $P'(\mathbf{s}_1, \mathbf{s}_2) \leftarrow P_1(\mathbf{s}_1)\mathscr{S}_{[0,\infty)} P_2(\mathbf{s}_2) \wedge P_1(\mathbf{s}_1)$, and the rule obtained by replacing M in r with $\boxminus_t P'(\mathbf{s}_1, \mathbf{s}_2) \wedge \boxminus_{(0,t]} P_1(\mathbf{s}_1)$, where P' is a fresh predicate whose arity equals the sum of arities of P_1 and P_2.

In the case of atoms mentioning $\mathscr{U}$, as well as $\top$ or $\bot$, we proceed analogously. $\qquad\square$

In the remainder of this section, we fix a normalised program Π and a dataset $\mathscr{D}$, and let $\mathsf{ground}(\Pi, \mathscr{D})$ be the subset of $\mathsf{ground}(\Pi)$ mentioning only constants from Π and

$\mathscr{D}$. Then, $\mathsf{at}(\Pi, \mathscr{D})$ is the set consisting of all relational atoms in $\mathscr{D}$, all metric atoms in $\mathsf{ground}(\Pi, \mathscr{D})$, and all metric atoms of the forms $\boxminus_{[0,\infty)}M$ and $\boxplus_{[0,\infty)}M$, with M a relational atom mentioned in $\mathsf{ground}(\Pi, \mathscr{D})$.

We next define the notion of a *window* – a fragment of an HT-interpretation over a particular interval; such windows will serve as states of our automata.

Definition 5.2
A *window* is a tuple (ϱ, H, T, b), where ϱ is a closed (and hence bounded) interval, $b \in \{0, 1\}$, and H and T are sets of metric facts of the form $M@t$ satisfying the following conditions:

– $M \in \mathsf{at}(\Pi, \mathscr{D})$, $t \in \varrho$, and $H \subseteq T$;
– there exist interpretations $\mathfrak{H}$ and $\mathfrak{T}$ such that, for each $M \in \mathsf{at}(\Pi, \mathscr{D})$ and $t \in \varrho$,

– $M@t \in H$ if and only if $\mathfrak{H} \models M@t$, and
– $M@t \in T$ if and only if $\mathfrak{T} \models M@t$.

The window's *length* is the length of ϱ. Finally, we say that a window is *initial* if either $H = T$ and $b = 0$, or $H \neq T$ and $b = 1$.

Intuitively, a window (ϱ, H, T, b) is a fragment of an HT-interpretation $(\mathfrak{H}, \mathfrak{T})$ restricted to ϱ, where H and T describe facts holding within ϱ in $\mathfrak{H}$ and $\mathfrak{T}$, respectively. Windows will serve as states of the automata recognising word representations of specific HT-interpretations, and in this process the flag b is used to distinguish between stable and non-stable models; in particular, our automata will ensure that flag b is set to 1 in each state $\mathscr{W}$ of a run such that $H \neq T$ in either $\mathscr{W}$ or in some previous state of this run.

By definition, a window can be extended to an HT-interpretation. This HT-interpretation can be an HT-model of Π only if the window locally satisfies Π, which we define next.

Definition 5.3
A window (ϱ, H, T, b) *locally satisfies* Π if, for each rule of Form (1) in $\mathsf{ground}(\Pi)$ and each $t \in \varrho$, both of the following hold:

– $M@t \in H$ if $M_i@t \in H$ for each $i \in \{1, \ldots, k\}$ and $M_j@t \notin T$ for each $j \in \{k+1, \ldots, m\}$,
– $M@t \in T$ if $M_i@t \in T$ for each $i \in \{1, \ldots, k\}$ and $M_j@t \notin T$ for each $j \in \{k+1, \ldots, m\}$.

Next, given an initial window $\mathscr{W}_0$, we define automata $\mathscr{A}_{\mathscr{W}_0}^{\leftarrow}$ and $\mathscr{A}_{\mathscr{W}_0}^{\rightarrow}$, which will allow us to recognise HT-models of Π that extend $\mathscr{W}_0$. In particular, if $\mathscr{A}_{\mathscr{W}_0}^{\leftarrow}$ and $\mathscr{A}_{\mathscr{W}_0}^{\rightarrow}$ accept words $\overleftarrow{w}$ and $\overrightarrow{w}$ respectively, then we will be able to construct an HT-model extending $\mathscr{W}_0$, for which the part located to the left of $\mathscr{W}_0$ is described by $\overleftarrow{w}$, and the part to the right of $\mathscr{W}_0$ by $\overrightarrow{w}$.

Definition 5.4
Let $\mathscr{W}_0 = (\varrho_0, H_0, T_0, b_0)$ be an initial window locally satisfying Π. Then, $\mathscr{A}_{\mathscr{W}_0}^{\leftarrow} = (\mathscr{Q}, \Sigma, \delta, q_0, F)$ is the generalised nondeterministic Büchi automaton with the following components:

1. the states $\mathcal{Q}$ consist of all windows of the the same length as $\mathcal{W}_0$ locally satisfying Π;
2. the alphabet Σ is the set of all $\sigma \subseteq \mathsf{at}(\Pi, \mathcal{D})$;
3. the transition function $\delta : \mathcal{Q} \times \Sigma \to 2^{\mathcal{Q}}$ is such that $(\varrho', H', T', b') \in \delta\big((\varrho, H, T, b), \sigma\big)$ if

   - $\varrho' = [\varrho^- - 1, \varrho^+ - 1]$,
   - $M@t \in H'$ if and only if $M@t \in H$, for every $M \in \mathsf{at}(\Pi, \mathcal{D})$ and $t \in \varrho' \cap \varrho$,
   - $T' = \{M@t' \in T \mid t' \in \varrho'\} \cup \{M@(\varrho^- - 1) \mid M \in \sigma\}$, and
   - $b' = 1$ whenever $b = 1$ or $H' \neq T'$, and $b' = 0$ otherwise;

4. the initial state q_0 is $\mathcal{W}_0$;
5. the accepting condition F is the family of sets of states which contains, for every atom $\boxminus_{[0,\infty)} M \in \mathsf{at}(\Pi, \mathcal{D})$, the sets

   $$\{(\varrho, H, T, b) \in \mathcal{Q} \mid \text{there exists } t \in \varrho \text{ such that } \boxminus_{[0,\infty)} M@t \in H \text{ or } M@t \notin H\},$$
   $$\{(\varrho, H, T, b) \in \mathcal{Q} \mid \text{there exists } t \in \varrho \text{ such that } \boxminus_{[0,\infty)} M@t \in T \text{ or } M@t \notin T\},$$

   and, for each $M_1 \mathcal{S}_{[0,\infty)} M_2 \in \mathsf{at}(\Pi, \mathcal{D})$, the sets

   $$\{(\varrho, H, T, b) \in \mathcal{Q} \mid \text{there exists } t \in \varrho \text{ such that } M_1 \mathcal{S}_{[0,\infty)} M_2@t \notin H \text{ or } M_2@t \in H\},$$
   $$\{(\varrho, H, T, b) \in \mathcal{Q} \mid \text{there exists } t \in \varrho \text{ such that } M_1 \mathcal{S}_{[0,\infty)} M_2@t \notin T \text{ or } M_2@t \in T\}.$$

The automaton $\mathscr{A}_{\mathcal{W}_0}^{\rightarrow}$ is defined analogously, except that we let $\varrho' = [\varrho^- + 1, \varrho^+ + 1]$, in the definition of T' we replace $\varrho^- - 1$ with $\varrho^+ + 1$, and in the definition of F we use $\boxplus$ and $\mathcal{U}$ instead of $\boxminus$ and $\mathcal{S}$, respectively.

Accepting runs of these automata will correspond to HT-interpretations. Indeed, as we will show next, each HT-interpretation can be decomposed into an infinite sequence of windows $\ldots, \mathcal{W}_{-1}, \mathcal{W}_0, \mathcal{W}_1, \ldots$ such that $\mathcal{W}_0, \mathcal{W}_{-1}, \ldots$ and $\mathcal{W}_0, \mathcal{W}_1, \ldots$ are accepting runs of $\mathscr{A}_{\mathcal{W}_0}^{\leftarrow}$ and $\mathscr{A}_{\mathcal{W}_0}^{\rightarrow}$, respectively. We define the decomposition of an HT-interpretation as follows.

Definition 5.5
We define the ϱ-*decomposition* of an HT-interpretation $(\mathfrak{H}, \mathfrak{T})$, for a bounded interval ϱ, as the sequence of tuples $\mathcal{W}_i = (\varrho_i, H_i, T_i, b_i)$, for $i \in \mathbb{Z}$, such that the following hold:

- $\varrho_i = [\varrho^- + i, \varrho^+ + i]$,
- H_i is the set of all facts $M@t$ such that $M \in \mathsf{at}(\Pi, \mathcal{D})$, $t \in \varrho_i$, and $\mathfrak{H} \models M@t$,
- T_i is the set of all facts $M@t$ such that $M \in \mathsf{at}(\Pi, \mathcal{D})$, $t \in \varrho_i$, and $\mathfrak{T} \models M@t$,
- $b_i = 1$ if there exists $j \in \{0, \ldots, i\}$ such that $H_j \neq T_j$; otherwise, $b_i = 0$.

Lemma 5.6
Let ϱ be a bounded interval, let a sequence of tuples $\mathcal{W}_i = (\varrho_i, H_i, T_i, b_i)$, with $i \in \mathbb{Z}$, be a ϱ-*decomposition* of an HT-model $(\mathfrak{H}, \mathfrak{T})$ of Π, and let $\overleftarrow{w} = \sigma_{-1} \sigma_{-2} \cdots$ and $\overrightarrow{w} = \sigma_1 \sigma_2 \cdots$ be the words such that $\sigma_k = T_k \setminus T_{k+1}$ for $k < 0$ and $\sigma_k = T_k \setminus T_{k-1}$ for $k > 0$. Then the following hold:

1. each $\mathcal{W}_i$ is a window locally satisfying Π,
2. $\mathcal{W}_0, \mathcal{W}_{-1}, \ldots$ is an accepting run of $\mathscr{A}_{\mathcal{W}_0}^{\leftarrow}$ on $\overleftarrow{w}$, and
3. $\mathcal{W}_0, \mathcal{W}_1, \ldots$ is an accepting run of $\mathscr{A}_{\mathcal{W}_0}^{\rightarrow}$ on $\overrightarrow{w}$.

Proof

To show that Statement 1 holds, we start by observing that each $\mathscr{W}_i$ satisfies all the conditions from Definition 5.2 of a window. Indeed, by Definition 5.5, H_i and T_i are sets of metric facts $M@t$ with $M \in \mathrm{at}(\Pi, \mathscr{D})$ and $t \in \varrho_i$ such that $H_i \subseteq T_i$, whereas $b_i \in \{0,1\}$; moreover, $\mathfrak{H}$ and $\mathfrak{T}$ witness existence of the interpretations required in Definition 5.5. Furthermore, since $(\mathfrak{H}, \mathfrak{T})$ is an HT-model of Π, each $\mathscr{W}_i$ locally satisfies Π.

To prove that Statement 2, we observe that $\mathscr{W}_0$ is an initial window by construction, so the automaton $\mathscr{A}_{\mathscr{W}_0}^{\leftarrow} = (\mathscr{Q}, \Sigma, \delta, q_0, F)$ is well-defined. By the definition of the transition function δ of $\mathscr{A}_{\mathscr{W}_0}^{\leftarrow}$ (see Definition 5.4) as well as by the construction of $\mathscr{W}_0, \mathscr{W}_{-1}, \dots$ and $\overleftarrow{w} = \sigma_{-1}\sigma_{-2}\dots$, we get that $\mathscr{W}_{i-1} \in \delta(\mathscr{W}_i, \sigma_{i-1})$, for each integer $i \leq 0$. Thus, $\mathscr{W}_0, \mathscr{W}_{-1}, \dots$ is a run of $\mathscr{A}_{\mathscr{W}_0}^{\leftarrow}$ on $\overleftarrow{w}$. It remains to show that this run is accepting, that is, for every set S in the accepting condition F, there are infinitely many integers $i < 0$ such that $\mathscr{W}_i \in S$. Towards a contradiction suppose that there exists $S \in F$ which does not satisfy this property. Thus, there exists $i \leq 0$ such that $\mathscr{W}_j \notin S$, for all $j \leq i$. Assume first that S is of one of the following forms

$$\{(\varrho', H, T, b) \in \mathscr{Q} \mid \text{there exists } t \in \varrho' \text{ such that } \boxminus_{[0,\infty)} M@t \in H \text{ or } M@t \notin H\}, \text{ or}$$

$$\{(\varrho', H, T, b) \in \mathscr{Q} \mid \text{there exists } t \in \varrho' \text{ such that } \boxminus_{[0,\infty)} M@t \in T \text{ or } M@t \notin T\},$$

for some $\boxminus_{[0,\infty)} M \in \mathrm{at}(\Pi, \mathscr{D})$. As a result, $\mathfrak{I} \not\models \boxminus_{[0,\infty)} M@t$ and $\mathfrak{I} \models M@t$, for each $t \leq \varrho_i^-$, where the interpretation $\mathfrak{I}$ is either $\mathfrak{H}$ or $\mathfrak{T}$, depending whether S is of the first or the second form presented above. This, however, contradicts the semantics of $\boxminus_{[0,\infty)}$. It remains to consider the case where S is of one of the following forms

$$\{(\varrho', H, T, b) \in \mathscr{Q} \mid \text{there exists } t \in \varrho' \text{ such that } M_1 \mathscr{S}_{[0,\infty)} M_2@t \notin H \text{ or } M_2@t \in H\}, \text{ or}$$

$$\{(\varrho', H, T, b) \in \mathscr{Q} \mid \text{there exists } t \in \varrho' \text{ such that } M_1 \mathscr{S}_{[0,\infty)} M_2@t \notin T \text{ or } M_2@t \in T\},$$

for some $M_1 \mathscr{S}_{[0,\infty)} M_2 \in \mathrm{at}(\Pi, \mathscr{D})$. Then, $\mathfrak{I} \models M_1 \mathscr{S}_{[0,\infty)} M_2@t$ and $\mathfrak{I} \not\models M_2@t$, for each $t \leq \varrho_i^-$, where $\mathfrak{I}$ is either $\mathfrak{H}$ or $\mathfrak{T}$, depending whether S is of the first or the second form presented above. This directly contradicts the semantics of $\mathscr{S}_{[0,\infty)}$. Consequently, $\mathscr{W}_0, \mathscr{W}_{-1}, \dots$ needs to be an accepting run of $\mathscr{A}_{\mathscr{W}_0}^{\leftarrow}$ on $\overleftarrow{w}$.

The proof of Statement 3 is analogous to the proof of Statement 2, due to the symmetry between the automata $\mathscr{A}_{\mathscr{W}_0}^{\leftarrow}$ and $\mathscr{A}_{\mathscr{W}_0}^{\rightarrow}$. □

To check existence of a stable model, however, we require automata that recognise HT-models $(\mathfrak{H}, \mathfrak{T})$ with $\mathfrak{H} = \mathfrak{T}$, and automata that recognise HT-models $(\mathfrak{H}, \mathfrak{T})$ with $\mathfrak{H} \subsetneq \mathfrak{T}$. The intersection of the former with the complement of the latter allows us to recognise stable models – that is, essentially, HT-models $(\mathfrak{T}, \mathfrak{T})$ for which there are no models $(\mathfrak{H}, \mathfrak{T})$ with $\mathfrak{H} \subsetneq \mathfrak{T}$. To this end, we define two more types of automata as follows:

Definition 5.7

Let $\mathscr{W}_0 = (\varrho_0, H_0, T_0, b_0)$ be an initial window locally satisfying Π. We define non-deterministic generalised Büchi automata $\mathscr{B}_{\mathscr{W}_0}^{\leftarrow}$, $\mathscr{B}_{\mathscr{W}_0}^{\rightarrow}$ and $\mathscr{C}_{\mathscr{W}_0}^{\leftarrow}$, $\mathscr{C}_{\mathscr{W}_0}^{\rightarrow}$ as follows:

- if $H_0 = T_0$ and $b_0 = 0$, the automata $\mathscr{B}_{\mathscr{W}_0}^{\leftarrow}$ and $\mathscr{B}_{\mathscr{W}_0}^{\rightarrow}$ are defined as $\mathscr{A}_{\mathscr{W}_0}^{\leftarrow}$ and $\mathscr{A}_{\mathscr{W}_0}^{\rightarrow}$, respectively, except that for a window (ϱ, H, T, b) to be a state we additionally require that $H = T$,
- the automata $\mathscr{C}_{\mathscr{W}_0}^{\leftarrow}$ and $\mathscr{C}_{\mathscr{W}_0}^{\rightarrow}$ are defined as $\mathscr{A}_{\mathscr{W}_0}^{\leftarrow}$ and $\mathscr{A}_{\mathscr{W}_0}^{\rightarrow}$, respectively, except that we add to the accepting condition the set $\{(\varrho, H, T, b) \in \mathscr{Q} \mid b = 1\}$.

Intuitively, if $\mathscr{W}_0$ satisfies $\mathscr{D}$, then the automata $\mathscr{B}_{\mathscr{W}_0}^{\leftarrow}$ and $\mathscr{B}_{\mathscr{W}_0}^{\rightarrow}$ recognise interpretations $\mathfrak{T}$ such that $(\mathfrak{T}, \mathfrak{T})$ is an HT-model of Π and $\mathscr{D}$. Furthermore, interpretations $\mathfrak{T}$ accepted by $\mathscr{A}_{\mathscr{W}_0}^{\leftarrow}$ and $\mathscr{C}_{\mathscr{W}_0}^{\rightarrow}$, or by $\mathscr{C}_{\mathscr{W}_0}^{\leftarrow}$ and $\mathscr{A}_{\mathscr{W}_0}^{\rightarrow}$ are such that $(\mathfrak{H}, \mathfrak{T})$ is an HT-model of Π and $\mathscr{D}$, for some $\mathfrak{H} \subsetneq \mathfrak{T}$. Hence, as we show next, we can use these automata to recognise stable models. This, however, requires the additional assumption that the windows in the automata are long enough to allow for capturing the semantics of metric operators occurring in a program. To this end, we will use windows of the same length as the interval $\varrho_{(\Pi,\mathscr{D})} = [t_{\mathscr{D}}^{\min}, t_{\mathscr{D}}^{\max} + t_\Pi]$. If the length of an initial window $\mathscr{W}_0$ is as required, then we can reduce checking existence of a stable model to checking particular properties of our automata, as stated next.

Theorem 5.8
Program Π and dataset $\mathscr{D}$ have a stable model if and only if there exists an initial window $\mathscr{W}_0 = (\varrho_0, T_0, T_0, 0)$ locally satisfying Π with $\varrho_0 = \varrho_{(\Pi,\mathscr{D})}$ and $T_0 \models \mathscr{D}$, and there exist words $\overleftarrow{w}$ and $\overrightarrow{w}$ over $2^{\mathsf{at}(\Pi,\mathscr{D})}$ such that both of the following hold:

1. $\overleftarrow{w}$ and $\overrightarrow{w}$ are accepted by $\mathscr{B}_{\mathscr{W}_0}^{\leftarrow}$ and $\mathscr{B}_{\mathscr{W}_0}^{\rightarrow}$, respectively,
2. there is no initial window $\mathscr{W}_0' = (\varrho_0, H_0, T_0, b_0)$ locally satisfying Π such that $H_0 \models \mathscr{D}$, and $\overleftarrow{w}$ and $\overrightarrow{w}$ are accepted either by $\mathscr{C}_{\mathscr{W}_0'}^{\leftarrow}$ and $\mathscr{A}_{\mathscr{W}_0'}^{\rightarrow}$, respectively, or by $\mathscr{A}_{\mathscr{W}_0'}^{\leftarrow}$ and $\mathscr{C}_{\mathscr{W}_0'}^{\rightarrow}$, respectively.

Proof
Assume that $\mathfrak{T}$ is a stable model of Π and $\mathscr{D}$. We will show how to construct the required $\mathscr{W}_0$, $\overleftarrow{w}$, and $\overrightarrow{w}$. To this end, let $\ldots, \mathscr{W}_{-1}, \mathscr{W}_0, \mathscr{W}_1, \ldots$ be the $\varrho_{(\Pi,\mathscr{D})}$-decomposition of $(\mathfrak{T}, \mathfrak{T})$ with $\mathscr{W}_i = (\varrho_i, H_i, T_i, b_i)$, for each $i \in \mathbb{Z}$. By Definition 5.5, we obtain that $H_i = T_i$ and $b_i = 0$, for all $i \in \mathbb{Z}$; furthermore, $\mathscr{W}_0$ locally satisfies Π by Lemma 5.6. Finally, we have $\varrho_0 = \varrho_{(\Pi,\mathscr{D})}$, which ensures $T_0 \models \mathscr{D}$, so $\mathscr{W}_0$ satisfies the initial requirements from the theorem. Next, let $\overleftarrow{w} = \sigma_{-1}\sigma_{-2}\cdots$ and $\overrightarrow{w} = \sigma_1\sigma_2\cdots$ be the words such that $\sigma_k = T_k \setminus T_{k+1}$ if $k < 0$, and $\sigma_k = T_k \setminus T_{k-1}$ if $k > 0$. It remains to show that $\mathscr{W}_0$, $\overleftarrow{w}$, and $\overrightarrow{w}$ satisfy Conditions 1 and 2 from the theorem.

To show that Condition 1 holds, we observe that, by Lemma 5.6, $\mathscr{W}_0, \mathscr{W}_{-1}, \ldots$ is an accepting run of $\mathscr{A}_{\mathscr{W}_0}^{\leftarrow}$ on $\overleftarrow{w}$, and $\mathscr{W}_0, \mathscr{W}_1, \ldots$ is an accepting run of $\mathscr{A}_{\mathscr{W}_0}^{\rightarrow}$ on $\overrightarrow{w}$. Moreover, since $H_i = T_i$ and $b_i = 0$ for all $i \in \mathbb{Z}$, we obtain that $\mathscr{W}_0, \mathscr{W}_{-1}, \ldots$ is an accepting run of $\mathscr{B}_{\mathscr{W}_0}^{\leftarrow}$ on $\overleftarrow{w}$, and $\mathscr{W}_0, \mathscr{W}_1, \ldots$ is an accepting run of $\mathscr{B}_{\mathscr{W}_0}^{\rightarrow}$ on $\overrightarrow{w}$.

Next, let us suppose towards a contradiction that there exists $\mathscr{W}_0' = (\varrho_0', H_0', T_0', b_0')$ witnessing a violation of Condition 2. Hence, $\mathscr{W}_0'$ is an initial window locally satisfying Π such that $\varrho_0' = \varrho_0$, $T_0' = T_0$, and $H_0' \models \mathscr{D}$. Moreover, we assume that $\overleftarrow{w}$ and $\overrightarrow{w}$ are accepted by $\mathscr{C}_{\mathscr{W}_0'}^{\leftarrow}$ and $\mathscr{A}_{\mathscr{W}_0'}^{\rightarrow}$, respectively. Hence, $\mathscr{C}_{\mathscr{W}_0'}^{\leftarrow}$ has an accepting run $\mathscr{W}_0', \mathscr{W}_{-1}', \ldots$ on $\overleftarrow{w}$ and $\mathscr{A}_{\mathscr{W}_0'}^{\rightarrow}$ has an accepting run $\mathscr{W}_0', \mathscr{W}_1', \ldots$ on $\overrightarrow{w}$, where we let $\mathscr{W}_i' = (\varrho_i', H_i', T_i', b_i')$, for all $i \in \mathbb{Z}$. Clearly, $\varrho_i' = \varrho_i$, for all $i \in \mathbb{Z}$. Moreover, by the definition of the transition functions of the automata and the construction of $\overleftarrow{w}$ and $\overrightarrow{w}$, we obtain that $T_i' = T_i$, for all $i \in \mathbb{Z}$. Therefore, $\mathfrak{T}$ is the least model of all relational facts in $\bigcup_{i \in \mathbb{Z}} T_i'$. We let $\mathfrak{H}$ be the least model of all relational facts in $\bigcup_{i \in \mathbb{Z}} H_i$; we will show that $(\mathfrak{H}, \mathfrak{T})$ is an HT-model of Π and $\mathscr{D}$.

Since $H_i' \subseteq T_i'$, for all $i \in \mathbb{Z}$, we obtain that $\mathfrak{H} \subseteq \mathfrak{T}$, and so $(\mathfrak{H}, \mathfrak{T})$ is an HT-interpretation. Moreover, $(\mathfrak{H}, \mathfrak{T})$ is an HT-model of $\mathscr{D}$, as $H_0' \models \mathscr{D}$. Next we will show that $(\mathfrak{H}, \mathfrak{T})$ is an HT-model of Π. Since each $\mathscr{W}_i'$ is a window of length $[t_{\mathscr{D}}^{\min}, t_{\mathscr{D}}^{\max} + t_\Pi]$

(which is the length of $\varrho_{(\Pi,\mathscr{D})}$), it holds that $\mathfrak{H} \models M@t$ if and only if $M@t \in H_i'$; as well as $\mathfrak{T} \models M@t$ if and only if $M@t \in T_i'$, for any $M \in \mathsf{at}(\Pi,\mathscr{D})$ and $t \in \varrho_i'$. Indeed, we have shown an analogous statement for positive (Wałęga *et al.* 2019, Lemma 9) and stratifiable programs (Tena Cucala *et al.* 2021), and the same argument applies here. Now, to show that $(\mathfrak{H},\mathfrak{T})$ is a model of Π, we fix a ground rule from $\mathsf{ground}(\Pi,\mathscr{D})$ of Form (1) and a time point t. If $\mathfrak{H}, t \models M_i$ for all $i \in \{1,\ldots,k\}$ and $\mathfrak{T}, t \not\models M_j$ for all $j \in \{k+1,\ldots,m\}$, then $M_i@t \in H_n'$ for all $i \in \{1,\ldots,k\}$ and $M_j@t \notin T_n$ for all $j \in \{k+1,\ldots,m\}$, for each n such that $t \in \varrho_n'$. Since $\mathscr{W}_n'$ is a window locally satisfying Π, we obtain that $M@t \in H_n$ (where M is the head of r), which implies $\mathfrak{H}, t \models M$ by definition of $\mathfrak{H}$ and the fact that Π is in normal form so M is a relational atom. Similarly, if $\mathfrak{T}, t \models M_i$ for all $i \in \{1,\ldots,k\}$ and $\mathfrak{T}, t \not\models M_j$ for all $j \in \{k+1,\ldots,m\}$, then we obtain that $\mathfrak{T}, t \models M$. Hence, $(\mathfrak{H},\mathfrak{T})$ is indeed an HT-model of Π.

However, the accepting condition of $\mathscr{C}_{\mathscr{W}_0'}^{\leftarrow}$ guarantees that $b_i' = 1$, for some $i \leq 0$, and so $H_i' \subsetneq T_i'$. Therefore, $\mathfrak{H} \subsetneq \mathfrak{T}$, and so $\mathfrak{T}$ is not a stable model, which rises a contradiction. If $\overleftarrow{w}$ and $\overrightarrow{w}$ are accepted by $\mathscr{A}_{\mathscr{W}_0'}^{\leftarrow}$ and $\mathscr{C}_{\mathscr{W}_0'}^{\rightarrow}$, respectively, then we construct in an analogous way a run $\mathscr{W}_0', \mathscr{W}_{-1}', \ldots$ of $\mathscr{A}_{\mathscr{W}_0'}^{\leftarrow}$ and a run $\mathscr{W}_0', \mathscr{W}_1', \ldots$ of $\mathscr{C}_{\mathscr{W}_0'}^{\rightarrow}$. Repeating the argumentation above, we construct an HT-model $(\mathfrak{H},\mathfrak{T})$ of Π and $\mathscr{D}$, and then, we show that there exists $i \geq 0$ such that $b_i' = 1$. Thus, $H_i' \subsetneq T_i'$, so $\mathfrak{H} \subsetneq \mathfrak{T}$, and consequently, $\mathfrak{T}$ is not a stable model, raising again a contradiction. Thus, Condition 2 holds.

For the converse implication, assume that there exist $\mathscr{W}_0$, $\overleftarrow{w}$, and $\overrightarrow{w}$ as described in the statement of the theorem. By Condition 1, there is an accepting run $\mathscr{W}_0, \mathscr{W}_{-1}, \ldots$ of $\mathscr{B}_{\mathscr{W}_0}^{\leftarrow}$ on $\overleftarrow{w}$, and an accepting run $\mathscr{W}_0, \mathscr{W}_1, \ldots$ of $\mathscr{B}_{\mathscr{W}_0}^{\rightarrow}$ on $\overrightarrow{w}$, where $\mathscr{W}_i = (\varrho_i, T_i, T_i, 0)$. We argue that the least model $\mathfrak{T}$ of relational facts in $\bigcup_{i\in\mathbb{Z}} T_i$ is a stable model of Π and $\mathscr{D}$. By an argument analogous to the second to last paragraph above, $(\mathfrak{T},\mathfrak{T})$ is an HT-model of Π and $\mathscr{D}$. Suppose for contradiction that $\mathfrak{T}$ is not stable, so there exists $\mathfrak{H} \subsetneq \mathfrak{T}$ such that $(\mathfrak{H},\mathfrak{T})$ is an HT-model of Π and $\mathscr{D}$. Let $\ldots, \mathscr{W}_{-1}', \mathscr{W}_0', \mathscr{W}_1', \ldots$ be the $\varrho_{(\Pi,\mathscr{D})}$-decomposition of $(\mathfrak{H},\mathfrak{T})$, with $\mathscr{W}_i' = (\varrho_i', H_i', T_i', b_i')$, and observe that $\mathscr{W}_0'$ is an initial window. By Lemma 5.6, $\mathscr{W}_0', \mathscr{W}_{-1}', \ldots$ is an accepting run of $\mathscr{A}_{\mathscr{W}_0'}^{\leftarrow}$ on $\overleftarrow{w}$, and $\mathscr{W}_0', \mathscr{W}_1', \ldots$ is an accepting run of $\mathscr{A}_{\mathscr{W}_0'}^{\rightarrow}$ on $\overrightarrow{w}$. Moreover, since $\mathfrak{H} \subsetneq \mathfrak{T}$, there is $i \in \mathbb{Z}$ such that $H_i' \neq T_i'$, and so $b_i' = 1$. If $i \leq 0$, then $b_j' = 1$ for all $j \leq i$, and so $\mathscr{C}_{\mathscr{W}_0'}^{\leftarrow}$ accepts $\overleftarrow{w}$; analogously, if $i \geq 0$, then $\mathscr{C}_{\mathscr{W}_0'}^{\rightarrow}$ accepts $\overrightarrow{w}$. Thus, Condition 2 does not hold, leading to a contradiction. $\square$

Theorem 5.8 reduces checking existence of a stable model to checking specific properties of our automata. We aim at showing that the latter is feasible in ExpSpace. The main obstacle, however, is the size of states in the automata: $\mathscr{W}_0$ is exponential in size with respect to $\mathscr{D}$, and states of the automata from Theorem 5.8 can be arbitrarily large since time points in windows can be arbitrary integers. To remedy the first issue, we let t_Π be the largest positive number mentioned in Π, and we let $t_\Pi = 1$ if Π mentions no positive numbers – this choice of value is arbitrary since in this case we only need t_Π to be a positive number in the timeline. Then, we show in Lemma 5.10 that it suffices to consider automata with states (i.e., windows) of length t_Π, which does not depend on $\mathscr{D}$. The second issue is addressed by Lemma 5.13 which tells us that, rather than considering automata with states of unbounded size (each of length t_Π), we can construct equivalent automata with polynomial-size states instead.

To state Lemma 5.10, we define the left-most and right-most fragments of length t_Π of a window as follows.

Definition 5.9
For a window $\mathscr{W} = (\varrho, H, T, b)$ of length at least t_Π, we define $\mathscr{W}^L = (\varrho^L, H^L, T^L, b^L)$, where

- $\varrho^L = [\varrho^-, \varrho^- + t_\Pi]$,
- $H^L = \{M@t \in H \mid t \in \varrho^L\}$,
- $T^L = \{M@t \in T \mid t \in \varrho^L\}$,
- $b^L = 1$ if $H^L \neq T^L$; otherwise, $b^L = 0$.

Analogously, we let $\mathscr{W}^R = (\varrho^R, H^R, T^R, b^R)$, where

- $\varrho^R = [\varrho^+ - t_\Pi, \varrho^+]$,
- $H^R = \{M@t \in H \mid t \in \varrho^R\}$,
- $T^R = \{M@t \in T \mid t \in \varrho^R\}$,
- $b^R = 1$ if $H^R \neq T^R$; otherwise, $b^R = 0$.

We observe that if $\mathscr{W}$ is a window locally satisfying Π, then $\mathscr{W}^L$ and $\mathscr{W}^R$ (which are "fragments" of $\mathscr{W}$) also are windows locally satisfying Π. Furthermore, both $\mathscr{W}^L$ and $\mathscr{W}^R$ are initial windows by definition. Thus, if $\mathscr{W}_0$ is such that $X_{\mathscr{W}_0}^{\leftarrow}$ and $X_{\mathscr{W}_0}^{\rightarrow}$ are well-defined automata, for some $X \in \{\mathscr{A}, \mathscr{B}, \mathscr{C}\}$, then $X_{\mathscr{W}_0^L}^{\leftarrow}$ and $X_{\mathscr{W}_0^R}^{\rightarrow}$ are also well-defined automata. Moreover, we can show several equivalences between these automata.

Lemma 5.10
Let $X \in \{\mathscr{A}, \mathscr{B}, \mathscr{C}\}$ and let $\mathscr{W}_0 = (\varrho_0, H_0, T_0, b_0)$ be a window such that $X_{\mathscr{W}_0}^{\leftarrow}$ and $X_{\mathscr{W}_0}^{\rightarrow}$ are well-defined. Then, the following statements hold:

1. If $X \in \{\mathscr{A}, \mathscr{B}\}$, then $X_{\mathscr{W}_0}^{\leftarrow}$ and $X_{\mathscr{W}_0}^{\rightarrow}$ are equivalent to $X_{\mathscr{W}_0^L}^{\leftarrow}$ and $X_{\mathscr{W}_0^R}^{\rightarrow}$, respectively.
2. If $X = \mathscr{C}$ and $H_0 = T_0$, then $X_{\mathscr{W}_0}^{\leftarrow}$ and $X_{\mathscr{W}_0}^{\rightarrow}$ are equivalent to $X_{\mathscr{W}_0^L}^{\leftarrow}$ and $X_{\mathscr{W}_0^R}^{\rightarrow}$, respectively.
3. If $X = \mathscr{C}$ and $H_0 \neq T_0$, then $X_{\mathscr{W}_0}^{\leftarrow}$ and $X_{\mathscr{W}_0}^{\rightarrow}$ are equivalent to $\mathscr{A}_{\mathscr{W}_0^L}^{\leftarrow}$ and $\mathscr{A}_{\mathscr{W}_0^R}^{\rightarrow}$, respectively.

Proof
First, we show Statement 1 for $X = \mathscr{A}$. Assume that $\mathscr{W}_0, \mathscr{W}_{-1}, \ldots$ is an accepting run of $\mathscr{A}_{\mathscr{W}_0}^{\leftarrow}$ on a word $w = \sigma_{-1}, \sigma_{-2}, \ldots$. We will show that there is an accepting run $\mathscr{W}_0', \mathscr{W}_{-1}', \ldots$ of $\mathscr{A}_{\mathscr{W}_0^L}^{\leftarrow}$ on the same word. To define this run, for each integer $i \leq 0$, we let $\mathscr{W}_i^L = (\varrho_i, H_i, T_i, b_i)$, and we define $\mathscr{W}_i' = (\varrho_i, H_i, T_i, b_i')$, where $b_i' = 1$ if there exists $j \in \{0, \ldots, i\}$ such that $H_i \neq T_i$, and otherwise $b_i' = 0$. In particular, $\mathscr{W}_0' = \mathscr{W}_0^L$. Now, we will show that $\mathscr{W}_0', \mathscr{W}_{-1}', \ldots$ is an accepting run of $\mathscr{A}_{\mathscr{W}_0^L}^{\leftarrow}$ on w. We start by observing that since each $\mathscr{W}_i$ is a window locally satisfying Π and since $\mathscr{W}_i^L$ is a "fragment" of $\mathscr{W}_i$, then $\mathscr{W}_i'$ is also a window locally satisfying Π. Thus, $\mathscr{W}_0', \mathscr{W}_{-1}', \ldots$ are states of $\mathscr{A}_{\mathscr{W}_0^L}^{\leftarrow}$. To show that they constitute a run of $\mathscr{A}_{\mathscr{W}_0^L}^{\leftarrow}$ on w, we observe that the transition functions δ of $\mathscr{A}_{\mathscr{W}_0}^{\leftarrow}$ and δ^L of $\mathscr{A}_{\mathscr{W}_0^L}^{\leftarrow}$ are the same modulo definition of states. This observation and our definition of b_i', ensure that $\mathscr{W}_{i-1} \in \delta(\mathscr{W}_i, \sigma_{i-1})$ implies $\mathscr{W}_{i-1}' \in \delta^L(\mathscr{W}_i', \sigma_{i-1})$, for any integer $i \leq 0$. It remains to show that $\mathscr{A}_{\mathscr{W}_0^L}^{\leftarrow}$ accepts the run $\mathscr{W}_0', \mathscr{W}_{-1}', \ldots$. For this, we observe that $\mathscr{A}_{\mathscr{W}_0}^{\leftarrow}$ and $\mathscr{A}_{\mathscr{W}_0^L}^{\leftarrow}$ have the same accepting conditions modulo the definition

of states. Next, let k be the difference between lengths of windows $\mathscr{W}_0$ and $\mathscr{W}_0^L$. Observe that for any integer $i \leq -k$, if $\mathscr{W}_i$ belongs to some set S from the accepting condition of $\mathscr{A}_{\mathscr{W}_0}^{\leftarrow}$, then there exists $j \in [i, i+k]$ such that $\mathscr{W}_j'$ belongs to a set from the accepting condition of $\mathscr{A}_{\mathscr{W}_0^L}^{\leftarrow}$ that corresponds to S. Since each set in the accepting condition of $\mathscr{A}_{\mathscr{W}_0}^{\leftarrow}$ is visited infinitely often by the states in the run $\mathscr{W}_0, \mathscr{W}_{-1}, \ldots$, each set in the accepting condition of $\mathscr{A}_{\mathscr{W}_0^L}^{\leftarrow}$ is also visited infinitely often by the states in the run $\mathscr{W}_0', \mathscr{W}_{-1}', \ldots$. Thus, the latter is an accepting run of $\mathscr{A}_{\mathscr{W}_0^L}^{\leftarrow}$. The case of $\mathscr{A}_{\mathscr{W}_0^R}^{\rightarrow}$ is symmetric so, by an analogous argumentation, we obtain that if $\mathscr{A}_{\mathscr{W}_0}^{\rightarrow}$ accepts a word w, then so does $\mathscr{A}_{\mathscr{W}_0^R}^{\rightarrow}$.

For the opposite direction of Statement 1, let us assume that $\mathscr{A}_{\mathscr{W}_0^L}^{\leftarrow}$ has an accepting run $\mathscr{W}_0^L, \mathscr{W}_{-1}', \mathscr{W}_{-2}', \ldots$ on a word $w = \sigma_{-1}, \sigma_{-2}, \ldots$; we will show that $\mathscr{A}_{\mathscr{W}_0}^{\leftarrow}$ has an accepting run on w. To facilitate the definition of this run, we divide $\mathscr{W}_0 = (\varrho_0, H_0, T_0, b_0)$ into tuples $\mathscr{W}_0', \ldots, \mathscr{W}_k'$, where k is the difference between the lengths of $\mathscr{W}_0$ and $\mathscr{W}_0^L$. In particular, for each $i \in \{0, \ldots, k\}$, we define $\mathscr{W}_i' = (\varrho_i', H_i', T_i', b_i')$, such that $\varrho_i' = [\varrho_0^- + k, \varrho_0^- + t_\Pi + k]$, $H_i' = \{M@t \in H_0 \mid t \in \varrho_i'\}$, $T_i' = \{M@t \in T_0 \mid t \in \varrho_i'\}$, whereas $b_i' = 1$ if $H_i' \neq T_i'$ and $b_i' = 0$ if $H_i' = T_i'$. Recall that since $\mathscr{A}_{\mathscr{W}_0}^{\leftarrow}$ is well-defined, $\mathscr{W}_0$ is an initial window locally satisfying Π, and as we already mentioned, by fragmenting windows locally satisfying Π we obtain windows which also locally satisfy Π. Hence all of $\mathscr{W}_0', \ldots, \mathscr{W}_k'$ locally satisfy Π. Furthermore, observe that $\mathscr{W}_0' = \mathscr{W}_0^L$. Next, we show how to merge the windows $\mathscr{W}_k', \ldots, \mathscr{W}_0', \mathscr{W}_{-1}', \ldots$ to obtain an accepting run of $\mathscr{A}_{\mathscr{W}_0}^{\leftarrow}$ on w. To this end, for arbitrary windows $\mathscr{W} = (\varrho, H, T, b)$ and $\mathscr{W}' = (\varrho', H', T', b')$ such that $\varrho \cap \varrho' \neq \emptyset$, we define their *union*, written as $\mathscr{W} \cup \mathscr{W}'$, as the tuple $(\varrho \cup \varrho', H \cup H', T \cup T', \max(b, b'))$. We can then establish the following claim:

Claim 5.11
Let $\mathscr{W} = (\varrho, H, T, b)$ and $\mathscr{W}' = (\varrho', H', T', b')$ be windows such that the left and right endpoints of ϱ' succeed those of ϱ, and for each $t \in \varrho \cap \varrho'$ and each $X \in \{H, T\}$, $M@t \in X$ if and only if $M@t \in X'$. If both $\mathscr{W}$ and $\mathscr{W}'$ are of length at least t_Π and locally satisfy Π, then $\mathscr{W} \cup \mathscr{W}'$ is a window locally satisfying Π.

Proof of Claim 5.11
Let $\mathscr{W}'' = \mathscr{W} \cup \mathscr{W}' = (\varrho'', H'', T'', b'')$. If $\mathscr{W}$ and $\mathscr{W}'$ locally satisfy Π, then clearly $\mathscr{W}''$ also locally satisfies Π. It remains to show that $\mathscr{W}''$ is a window. To this end, for $\mathfrak{I}$ an interpretation, ϱ_b a closed interval, and J a set of metric facts $M@t$ with $M \in \mathsf{at}(\Pi, \mathscr{D})$ and $t \in \varrho_b$, we say that $\mathfrak{I}$ *corresponds* to J over ϱ_b if, for every $t \in \varrho_b$ and $M \in \mathsf{at}(\Pi, \mathscr{D})$, it holds that $\mathfrak{I} \models M@t$ if and only if $M@t \in J$. To show that $\mathscr{W}''$ is a window, the only non-trivial condition that needs to be verified is that there exist two interpretations that correspond, respectively, to H'' and T'' over ϱ''. We show this for H'' only, as the argument for T'' is analogous. Let $\mathfrak{H}$ (resp. $\mathfrak{H}'$) be an interpretation corresponding to H over ϱ (resp. H' over ϱ'); such an interpretation exists, since $\mathscr{W}$ (resp. $\mathscr{W}'$) is a window. Let $\mathfrak{H}''$ be an arbitrary interpretation that coincides with $\mathfrak{H}$ over ϱ and all time points to its left, and with $\mathfrak{H}'$ over ϱ' and all time points to its right. Note that $\mathfrak{H}$ and $\mathfrak{H}'$ agree over $\varrho \cap \varrho'$ since, by assumption, $M@t \in H$ if and only if $M@t \in H'$ for all $t \in \varrho \cap \varrho'$; thus, $\mathfrak{H}''$ is well-defined. It remains to show that $\mathfrak{H}''$ corresponds to H'' over ϱ'' – that is, for each $M@t$ with $M \in \mathsf{at}(\Pi, \mathscr{D})$ and $t \in \varrho''$, we must show that $\mathfrak{H}'' \models M@t$ if and only if $M@t \in H''$. We show this explicitly for the case where M is of the form $\boxminus_{\varrho_b} P$, for ϱ_b a bounded interval and P a relational atom; the remaining cases can be proved in

a similar way; see Wałęga *et al.* (2019), Lemma 7. Suppose that $t \in \varrho$. Since $\mathfrak{H}''$ and $\mathfrak{H}$ coincide over ϱ and all time points to its left, we have that $\mathfrak{H}'' \models \boxminus_{\varrho_b} P@t$ if and only if $\mathfrak{H} \models \boxminus_{\varrho_b} P@t$. Then, since $\mathfrak{H}$ corresponds to H over ϱ, it holds that $\mathfrak{H} \models \boxminus_{\varrho_b} P@t$ if and only if $\boxminus_{\varrho_b} P@t \in H$. Finally, since H and H'' coincide over ϱ, $\boxminus_{\varrho_b} P@t \in H$ if and only if $\boxminus_{\varrho_b} P@t \in H''$. Thus, by chaining together these double implications, we have that $\mathfrak{H}'' \models \boxminus_{\varrho_b} P@t$ if and only if $\boxminus_{\varrho_b} P@t \in H''$, which is the target equivalence. Suppose now that $t \notin \varrho$, so t is the right endpoint of ϱ'. By the semantics of metric atoms, $\mathfrak{H}'' \models \boxminus_{\varrho_b} P@t$ if and only if $\mathfrak{H}'' \models P@t - t'$ for each $t' \in \varrho_b$. Since the length of ϱ' is at least t_Π, we have $t - t' \in \varrho'$ for each $t' \in \varrho_b$. Hence, since $\mathfrak{H}''$ and $\mathfrak{H}'$ coincide over ϱ', $\mathfrak{H}'' \models \boxminus_{\varrho_b} P$ if and only if $\mathfrak{H}' \models \boxminus_{\varrho_b} P$, and the rest follows by an argument analogous to the previous case, finishing the proof of Claim 5.11. $\qquad\square$

We resume the proof of Lemma 5.10. Now, we construct an accepting run of $\mathscr{A}^{\leftarrow}_{\mathscr{W}_0}$ on w as follows. For each integer $i < 0$, we let $\mathscr{W}_i = (\varrho_i, H_i, T_i, b_i)$ be $\mathscr{W}'_i \cup \cdots \cup \mathscr{W}'_{i+k}$ with the exception that $b_i = 1$ if there exists $j \in \{i, \ldots, k\}$ such that $H'_i \neq T'_i$ and $b_i = 0$ otherwise.

Now, we will show that $\mathscr{W}_0, \mathscr{W}_{-1}, \ldots$ is indeed an accepting run of $\mathscr{A}^{\leftarrow}_{\mathscr{W}_0}$ on w. To this end, we observe that each $\mathscr{W}_i$ is a window locally satisfying Π, since so is $\mathscr{W}'_i \cup \cdots \cup \mathscr{W}'_{i+k}$ by Claim 5.11; furthermore, each $\mathscr{W}_i$ it is of the same length as $\mathscr{W}_0$. Therefore, each $\mathscr{W}_i$ is a state of the automaton $\mathscr{A}^{\leftarrow}_{\mathscr{W}_0}$. Moreover, the definition of $\mathscr{W}_i$ ensures that if there exists a transition on σ_{i-1} from $\mathscr{W}'_i$ to $\mathscr{W}'_{i-1}$ in $\mathscr{A}^{\leftarrow}_{\mathscr{W}_0^L}$, then there exists a transition on σ_{i-1} from $\mathscr{W}_i$ to $\mathscr{W}_{i-1}$ in $\mathscr{A}^{\leftarrow}_{\mathscr{W}_0}$, so $\mathscr{W}_0, \mathscr{W}_{-1}, \ldots$ is a run of $\mathscr{A}^{\leftarrow}_{\mathscr{W}_0}$ on w. Furthermore, if $\mathscr{W}'_i$ belongs to some set S in the accepting condition of $\mathscr{A}^{\leftarrow}_{\mathscr{W}_0^L}$, then the bigger window $\mathscr{W}_i$ belongs to the set corresponding to S in the accepting condition of $\mathscr{A}^{\leftarrow}_{\mathscr{W}_0}$. Hence, $\mathscr{A}^{\leftarrow}_{\mathscr{W}_0}$ accepts the run $\mathscr{W}_0, \mathscr{W}_{-1}, \ldots$. The case of $\mathscr{A}^{\rightarrow}_{\mathscr{W}_0}$ is symmetric, so we obtain that if $\mathscr{A}^{\rightarrow}_{\mathscr{W}_0^R}$ accepts a word w, then so does $\mathscr{A}^{\rightarrow}_{\mathscr{W}_0}$.

To finish the proof of Statement 1, it remains to consider the case $X = \mathscr{B}$. Recall that the only difference between the automata of the types $\mathscr{A}$ and $\mathscr{B}$ is that the latter impose additional requirement on the windows to be states (see Definition 5.7), namely for a window $\mathscr{W} = (\varrho, H, T, b)$ to be a state, it is required that $H = T$. This, however, does not affect our argumentation above, and so, we obtain that $\mathscr{B}^{\leftarrow}_{\mathscr{W}_0}$ and $\mathscr{B}^{\rightarrow}_{\mathscr{W}_0}$ accept the same words as $\mathscr{B}^{\leftarrow}_{\mathscr{W}_0^L}$ and $\mathscr{B}^{\rightarrow}_{\mathscr{W}_0^R}$, respectively.

To show that Statements 2 and 3 hold, we recall that the automata $\mathscr{C}$ differ from automata $\mathscr{A}$ only in the existence of an additional set $S = \{(\varrho, H, T, b) \in \mathscr{Q} \mid b = 1\}$ in their accepting conditions (see Definition 5.7). If $\mathscr{W}_0 = (\varrho_0, H_0, T_0, b_0)$ is such that $H_0 = T_0$, then we can show that $\mathscr{C}^{\leftarrow}_{\mathscr{W}_0}$ and $\mathscr{C}^{\rightarrow}_{\mathscr{W}_0}$ accept the same words as $\mathscr{C}^{\leftarrow}_{\mathscr{W}_0^L}$ and $\mathscr{C}^{\rightarrow}_{\mathscr{W}_0^R}$, respectively, using the same argumentation as in the proof of Statement 1, except for the following difference. We use the fact that $H_0 = T_0$ to ensure that in an arbitrary accepting run of $\mathscr{C}^{\leftarrow}_{\mathscr{W}_0}$ for some word, there exists a window $\mathscr{W}_i = (\varrho_i, H_i, T_i, b_i)$ to the left of $\mathscr{W}_0$ with $H_i \neq T_i$, which in turn allows us to ensure that the corresponding run that we define for $\mathscr{C}^{\leftarrow}_{\mathscr{W}_0^L}$ contains a "fragment" $\mathscr{W}'_j = (\varrho'_j, H'_j, T'_j, b'_j)$ of $\mathscr{W}_i$ with $H'_j \neq T'_j$, and hence the run is accepting. An analogous argument applies to $\mathscr{C}^{\rightarrow}_{\mathscr{W}_0}$. Thus, Statement 2 holds.

However, if $\mathscr{W}_0 = (\varrho_0, H_0, T_0, b_0)$ is such that $H_0 \neq T_0$ – as in Statement 3 – then $b_0 = 1$ since $\mathscr{W}_0$ is initial, and so, each window in any run of the automata $\mathscr{C}^{\leftarrow}_{\mathscr{W}_0}$ and $\mathscr{C}^{\rightarrow}_{\mathscr{W}_0}$ will belong to the additional set $\{(\varrho, H, T, b) \in \mathscr{Q} \mid b = 1\}$ from the accepting condition.

Hence, this additional condition is trivially satisfied by any infinite run, so $\mathscr{C}^{\leftarrow}_{\mathscr{W}_0}$ and $\mathscr{C}^{\rightarrow}_{\mathscr{W}_0}$ are equivalent to $\mathscr{A}^{\leftarrow}_{\mathscr{W}_0}$ and $\mathscr{A}^{\rightarrow}_{\mathscr{W}_0}$, respectively. Therefore, by Statement 1, we obtain that $\mathscr{C}^{\leftarrow}_{\mathscr{W}_0}$ and $\mathscr{C}^{\rightarrow}_{\mathscr{W}_0}$ are also equivalent to $\mathscr{A}^{\leftarrow}_{\mathscr{W}_0^L}$ and $\mathscr{A}^{\rightarrow}_{\mathscr{W}_0^R}$, respectively, which proves Statement 3. □

Lemma 5.10 allows us to restrict our attention to automata with windows of polynomial length, but representations of such windows can still be unbounded, because arbitrarily large integers may occur in them. However, we will show in Lemma 5.13 that we can construct equivalent automata with states that can be represented in polynomial space, by merging "similar" states in the original automata. We use a notion of similarity given by the relation $\sim$ defined next:

Definition 5.12
Let $\sim$ be the equivalence relation on windows such that $\mathscr{W} \sim \mathscr{W}'$ if and only if $\mathscr{W}'$ is obtained by increasing all time points mentioned in $\mathscr{W}$ by some integer c. Let $[\mathscr{W}]_\sim$ be the equivalence class of $\mathscr{W}$ with respect to $\sim$. Then, for any $X \in \{\mathscr{A}, \mathscr{B}, \mathscr{C}\}$ and any initial window $\mathscr{W}_0$, if automaton $X^{\leftarrow}_{\mathscr{W}_0^L} = (\mathscr{Q}, \Sigma, \delta, q_0, F)$ is well-defined, then $\widetilde{X}^{\leftarrow}_{[\mathscr{W}_0^L]_\sim} = (\widetilde{\mathscr{Q}}, \widetilde{\Sigma}, \widetilde{\delta}, \widetilde{q}_0, \widetilde{F})$ is the nondeterministic Büchi automaton defined as follows:

1. $\widetilde{\mathscr{Q}}$ is the quotient set of $\mathscr{Q}$ by the relation $\sim$,
2. $\widetilde{\Sigma} = \Sigma$,
3. $\widetilde{\delta}$ is such that, for any $q, q' \in \widetilde{\mathscr{Q}}$ and $\sigma \in \Sigma$, it holds that $q' \in \widetilde{\delta}(q, \sigma)$ if and only if there exist $\mathscr{W}, \mathscr{W}' \in \mathscr{Q}$ such that $[\mathscr{W}]_\sim = q$, $[\mathscr{W}']_\sim = q'$, and $q' \in \delta(q, \sigma)$,
4. $\widetilde{q}_0 = [q_0]_\sim$,
5. $\widetilde{F} = \{S'_1, \ldots, S'_n\}$ is a family of subsets of $\widetilde{\mathscr{Q}}$ such that $q \in S'_i$ if and only if there exists $\mathscr{W} \in S_i$ such that $[\mathscr{W}]_\sim = q$;

and if $X^{\rightarrow}_{\mathscr{W}_0^R} = (\mathscr{Q}, \Sigma, \delta, q_0, F)$ is well-defined, then automaton $\widetilde{X}^{\rightarrow}_{[\mathscr{W}_0^R]_\sim}$ is defined analogously.

Now we show that the new automata $\widetilde{X}^{\leftarrow}_{[\mathscr{W}_0^L]_\sim}$ and $\widetilde{X}^{\rightarrow}_{[\mathscr{W}_0^L]_\sim}$ are equivalent to the previously defined automata $X^{\leftarrow}_{\mathscr{W}_0^L}$ an $X^{\rightarrow}_{\mathscr{W}_0^L}$, respectively.

Lemma 5.13
Let $X \in \{\mathscr{A}, \mathscr{B}, \mathscr{C}\}$ and let $\mathscr{W}_0 = (\varrho_0, H_0, T_0, b_0)$ be an initial window locally satisfying Π such that $X^{\leftarrow}_{\mathscr{W}_0}$ and $X^{\rightarrow}_{\mathscr{W}_0}$ are well-defined. Then, $X^{\leftarrow}_{\mathscr{W}_0^L}$ and $X^{\rightarrow}_{\mathscr{W}_0^R}$ are equivalent to $\widetilde{X}^{\leftarrow}_{[\mathscr{W}_0^L]_\sim}$ and $\widetilde{X}^{\rightarrow}_{[\mathscr{W}_0^R]_\sim}$, respectively.

Proof
Assume that $\mathscr{W}_0, \mathscr{W}_{-1}, \ldots$ is an accepting run of $X^{\leftarrow}_{\mathscr{W}_0^L}$ on a word w. Then, by Definition 5.12, $[\mathscr{W}_0]_\sim, [\mathscr{W}_{-1}]_\sim, \ldots$ are states of $\widetilde{X}^{\leftarrow}_{[\mathscr{W}_0^L]_\sim}$ and $[\mathscr{W}_0]_\sim = [\mathscr{W}_0^L]_\sim$. Moreover, by the definition of the transition function and the accepting conditions of $\widetilde{X}^{\leftarrow}_{[\mathscr{W}_0^L]_\sim}$, the fact that $\mathscr{W}_0, \mathscr{W}_{-1}, \ldots$ is an accepting run of $X^{\leftarrow}_{\mathscr{W}_0^L}$ on w implies that $[\mathscr{W}_0]_\sim, [\mathscr{W}_{-1}]_\sim, \ldots$ is an accepting run of $\widetilde{X}^{\leftarrow}_{[\mathscr{W}_0^L]_\sim}$ on the same word w. Similarly, we can show that if $X^{\rightarrow}_{\mathscr{W}_0^R}$ accepts w, then so does $\widetilde{X}^{\rightarrow}_{[\mathscr{W}_0^R]_\sim}$.

For the opposite direction, let us assume that $q_0, q_{-1}, \ldots$ is an accepting run of $\widetilde{X}^{\leftarrow}_{[\mathscr{W}_0^L]_\sim}$ on a word w. For each integer $i < 0$, we let $\mathscr{W}_i = (\varrho_i, H_i, T_i, b_i)$ be a state of $X^{\leftarrow}_{\mathscr{W}_0^L}$

such that $[\mathscr{W}_i]_\sim = q_i$ and $\varrho_i = [\varrho_0^- + i, \varrho_0^+ + i]$; such states are guaranteed to exist by definition of $X_{[\mathscr{W}_0^L]_\sim}^\leftarrow$. We show that $\mathscr{W}_0, \mathscr{W}_{-1}, \ldots$ is an accepting run of $X_{\mathscr{W}_0^L}^\leftarrow$ on w. To this end, we will show that for every integer $i \leq 0$ and every symbol σ, if $q_{i-1} \in \widetilde{\delta}(q_i, \sigma)$, then $\mathscr{W}_{i-1} \in \delta(\mathscr{W}_i, \sigma)$, where $\widetilde{\delta}$ and δ are the transition functions of $\widetilde{X}_{[\mathscr{W}_0^L]_\sim}^\leftarrow$ and $X_{\mathscr{W}_0^L}^\leftarrow$, respectively. Indeed, by definition, if $q_{i-1} \in \widetilde{\delta}(q_i, \sigma)$, then there are states $\mathscr{W}$ and $\mathscr{W}'$ of $X_{\mathscr{W}_0^L}^\leftarrow$ such that $[\mathscr{W}]_\sim = q_i$, $[\mathscr{W}']_\sim = q_{i-1}$, and $\mathscr{W}' \in \delta(\mathscr{W}, \sigma)$. Hence, $\mathscr{W}_i \sim \mathscr{W}$ and $\mathscr{W}_{i-1} \sim \mathscr{W}'$. This, by the definition of $\sim$ and the fact that $\mathscr{W}' \in \delta(\mathscr{W}, \sigma)$, implies that $\mathscr{W}_{i-1} \in \delta(\mathscr{W}_i, \sigma)$. Therefore, $\mathscr{W}_0, \mathscr{W}_{-1}, \ldots$ is a run of $X_{\mathscr{W}_0^L}^\leftarrow$ on w. Finally, since $q_0, q_{-1}, \ldots$ is an accepting run of $\widetilde{X}_{[\mathscr{W}_0^L]_\sim}^\leftarrow$, by the definition of accepting conditions, we obtain that $\mathscr{W}_0, \mathscr{W}_{-1}, \ldots$ is an accepting run of $X_{\mathscr{W}_0^L}^\leftarrow$. The same argumentation shows that if $\widetilde{X}_{[\mathscr{W}_0^R]_\sim}^\rightarrow$ accepts w, then $X_{\mathscr{W}_0^R}^\rightarrow$ also does so. □

We are now ready to show ExpSpace data complexity of reasoning in DatalogMTL$^\neg$ over $\mathbb{Z}$.

Theorem 5.14
Checking whether a DatalogMTL$^\neg$ program and a dataset have a stable model over the integer timeline is in ExpSpace with respect to data complexity.

Proof
It suffices to show that checking existence of $\mathscr{W}_0$, $\overleftarrow{w}$, and $\overrightarrow{w}$ from Theorem 5.8 is feasible in ExpSpace in the size of (the representation of) $\mathscr{D}$. First, we observe that the length of $\varrho_{(\Pi, \mathscr{D})}$ is exponential, and so is the representation of windows over $\varrho_0 = \varrho_{(\Pi, \mathscr{D})}$. Thus, it is feasible in ExpSpace to guess a tuple $\mathscr{W}_0 = (\varrho_0, T_0, T_0, 0)$ as well as to verify that it is an initial window locally satisfying Π and such that $T_0 \models \mathscr{D}$, as required in Theorem 5.8.

Thus, it remains to check existence of $\overleftarrow{w}$ and $\overrightarrow{w}$ satisfying Conditions 1 and 2 from Theorem 5.8. To this end, we will treat a candidate pair of words $w_1 = \sigma_{-1} \sigma_{-2} \cdots$ and $w_2 = \sigma_1 \sigma_2 \cdots$ as a single word $w_1 w_2 = (\sigma_{-1}, \sigma_1)(\sigma_{-2}, \sigma_2) \cdots$ and combine pairs of corresponding automata so that they accept combined words. In particular, for automata X and Y, we let XY be an automaton which on a word $w_1 w_2$ simulates the run of X on w_1 and the run of Y on w_2, simultaneously. Such an automaton XY is polynomially bigger than X and Y as its states are pairs consisting of a state of X and a state of Y. Thus, checking existence of $\overleftarrow{w}$ and $\overrightarrow{w}$ from Theorem 5.8 reduces to checking existence of $\overleftarrow{w}\,\overrightarrow{w}$ such that

(i) $\overleftarrow{w}\,\overrightarrow{w}$ is accepted by $\mathscr{B}_{\mathscr{W}_0}^\leftarrow \mathscr{B}_{\mathscr{W}_0}^\rightarrow$ (Condition 1) and
(ii) there exists no $\mathscr{W}_0'$ (satisfying properties from Condition 2) such that $\overleftarrow{w}\,\overrightarrow{w}$ is accepted by $\mathscr{C}_{\mathscr{W}_0'}^\leftarrow \mathscr{A}_{\mathscr{W}_0'}^\rightarrow$ or by $\mathscr{A}_{\mathscr{W}_0'}^\leftarrow \mathscr{C}_{\mathscr{W}_0'}^\rightarrow$.

By Lemmas 5.10 and 5.13, Item (i) reduces to checking if $\overleftarrow{w}\,\overrightarrow{w}$ is accepted by $\widetilde{\mathscr{B}}_{[\mathscr{W}_0^L]_\sim}^\leftarrow \widetilde{\mathscr{B}}_{[\mathscr{W}_0^R]_\sim}^\rightarrow$ and Item (ii) reduces to checking non-existence of $\mathscr{W}_0' = (\varrho_0, H_0, T_0, b)$ (satisfying properties from Condition 2) such that $\overleftarrow{w}\,\overrightarrow{w}$ is accepted by the union of the automata $\widetilde{\mathscr{C}}_{[(\mathscr{W}_0')^L]_\sim}^\leftarrow \widetilde{\mathscr{A}}_{[(\mathscr{W}_0)'^R]_\sim}^\rightarrow$ and $\widetilde{\mathscr{C}}_{[(\mathscr{W}_0)'^L]_\sim}^\leftarrow \widetilde{\mathscr{A}}_{[(\mathscr{W}_0')^R]_\sim}^\rightarrow$, if $H_0 = T_0$; or by the automaton $\widetilde{\mathscr{A}}_{[(\mathscr{W}_0')^L]_\sim}^\leftarrow \widetilde{\mathscr{A}}_{[(\mathscr{W}_0')^R]_\sim}^\rightarrow$, if $H_0 \neq T_0$.

To perform these checks in ExpSpace, we will characterise each initial window of the form $\mathscr{W}_0' = (\varrho_0, H_0, T_0, b_0)$ locally satisfying Π and such that $H_0 \models \mathscr{D}$ – as mentioned

in Condition 2 of Theorem 5.8 – by a triple consisting of two initial windows $(\mathscr{W}_0')^L$, $(\mathscr{W}_0')^R$, and a flag $b \in \{0, 1\}$ such that $b = 1$ if and only if $H_0 \neq T_0$ (so several windows can be characterised by the same triple). Now, we will show that a set $\mathscr{T}$ of all triples characterising all such windows $\mathscr{W}_0'$ can be constructed in ExpSpace. We observe that all $(\mathscr{W}_0')^L$ are over the same interval ϱ^L, which is independent from the choice of the window $\mathscr{W}_0'$, since all $\mathscr{W}_0'$ are over the same interval $\varrho_{(\Pi, \mathscr{D})}$. Similarly, $(\mathscr{W}_0')^R$ are over the same interval ϱ^R, which is independent from the choice of $\mathscr{W}_0'$. Moreover, these ϱ^L and ϱ^R are polynomially long, and so, each $(\mathscr{W}_0')^L$ and $(\mathscr{W}_0')^R$ has a polynomial representation. Thus, there are exponentially many triples $(\mathscr{W}_1, \mathscr{W}_2, b)$ which need to be checked for membership in $\mathscr{T}$. Our ExpSpace procedure iterates through all these triples. For each triple $(\mathscr{W}_1, \mathscr{W}_2, b)$, the procedure checks if there exists an initial window $\mathscr{W}_0' = (\varrho_0, H_0, T_0, b_0)$ locally satisfying Π such that $H_0 \models \mathscr{D}$, $(\mathscr{W}_0')^L = \mathscr{W}_1$, $(\mathscr{W}_0')^R = \mathscr{W}_2$, and $b = 1$ if and only if $H_0 \neq T_0$. All these checks are feasible in ExpSpace, and if they yield positive answers, then $(\mathscr{W}_1, \mathscr{W}_2, b) \in \mathscr{T}$.

We observe that checking Items (i) and (ii) reduces to checking non-emptiness of an automaton obtained by intersecting $\overleftarrow{\mathscr{B}}_{[\mathscr{W}_0^L]_\sim} \overrightarrow{\mathscr{B}}_{[\mathscr{W}_0^R]_\sim}$ with complements of all the automata corresponding to triples from $\mathscr{T}$, where we say that a triple $((\mathscr{W}_0')^L, (\mathscr{W}_0')^R, b) \in \mathscr{T}$ *corresponds* to the automaton which is the intersection of $\overleftarrow{\mathscr{C}}_{[(\mathscr{W}_0')^L]_\sim} \overrightarrow{\mathscr{A}}_{[(\mathscr{W}_0)'^R]_\sim}$ and $\overleftarrow{\mathscr{C}}_{[(\mathscr{W}_0)'^L]_\sim} \overrightarrow{\mathscr{A}}_{[(\mathscr{W}_0')^R]_\sim}$, if $b = 0$; and which is $\overleftarrow{\mathscr{A}}_{[(\mathscr{W}_0')^L]_\sim} \overrightarrow{\mathscr{A}}_{[(\mathscr{W}_0')^R]_\sim}$, if $b = 1$. To conclude, we show that this check is feasible in ExpSpace. Indeed, by construction, all automata mentioned above have states that can be represented in polynomial space. However, since these automata are nondeterministic, their complements have states of exponential size. To intersect the exponentially many such complemented automata, we construct an automaton whose states are exponentially long tuples, whose ith element is a state of the ith complemented automata. Such an exponentially long tuple of exponentially large states is itself exponentially representable. Thus, the obtained automaton has exponentially big states. Checking non-emptiness of this automaton is feasible in ExpSpace using the standard on-the-fly approach, where states are guessed one-by-one (Baier and Katoen 2008).   □

The procedure outlined in the proof of the theorem shows that reasoning in DatalogMTL$^{\neg}$ over the integer timeline is in ExpSpace in data complexity. In the next section, we show how restricting the form of DatalogMTL$^{\neg}$ programs to the forward-propagating fragment allows us to establish a tight PSpace bound for data complexity.

5.2 Forward-propagating programs

In this section, we consider reasoning with forward-propagating programs (see Definition 3.1) and bounded datasets. This setting has already been studied in the context of stream reasoning (Ronca *et al.* 2018; Wałęga *et al.* 2019).

The normalisation of a DatalogMTL$^{\neg}_{\mathsf{FP}}$ program, as defined in Section 5.1, results also in a DatalogMTL$^{\neg}_{\mathsf{FP}}$ program; thus, for the remainder of this section, we let Π be a fixed (but arbitrary) DatalogMTL$^{\neg}_{\mathsf{FP}}$ program in normal form and let $\mathscr{D}$ be a bounded dataset.

Restricting ourselves to forward-propagating programs and bounded datasets will enable a simplification of the procedure in Section 5.1, where we check existence of a stable

model of Π and $\mathscr{D}$ by looking for an initial window $\mathscr{W}_0$ and a pair of words $\overleftarrow{w}$, $\overrightarrow{w}$ satisfying the conditions in Theorem 5.8. Towards this goal, we first show in Lemma 5.16 that we can guess the initial window $\mathscr{W}_0$ over an interval located to the left of all intervals in $\mathscr{D}$, instead of $\varrho_{(\Pi,\mathscr{D})}$ as in the previous section. Furthermore, we can use the fact that Π is forward-propagating to show that checking existence of word $\overleftarrow{w}$ can be done independently of $\mathscr{D}$. Finally, to check existence of word $\overrightarrow{w}$, we define a new family of automata of the form $\mathscr{F}_{\mathscr{W}_0}^{\rightarrow}$, which can be used instead of $X_{\mathscr{W}_0}^{\rightarrow}$, for $X \in \{\mathscr{A}, \mathscr{B}, \mathscr{C}\}$; doing so requires no complementation, and thus we avoid the exponential blowup. As a result, the procedure becomes feasible in polynomial space.

We next define a new automaton $\mathscr{F}_{\mathscr{W}_0}^{\rightarrow}$ as a refinement of $\mathscr{B}_{\mathscr{W}_0}^{\rightarrow}$. Furthermore, we impose an additional restriction on states of $\mathscr{F}_{\mathscr{W}_0}^{\rightarrow}$ to guarantee their minimality, as described below.

Definition 5.15
We say that a window (ϱ, H, T, b) *locally satisfies* $\mathscr{D}$ if $M@t \in H$ for each $M \in \mathsf{at}(\Pi, \mathscr{D})$ and each $t \in \varrho$ such that $\mathscr{D} \models M@t$. Let $\mathscr{W}_0 = (\varrho_0, T_0, T_0, 0)$ be an initial window locally satisfying Π (see Definition 5.3) and $\mathscr{D}$. We define the generalised Büchi automaton $\mathscr{F}_{\mathscr{W}_0}^{\rightarrow}$ analogously to the automaton $\mathscr{B}_{\mathscr{W}_0}^{\rightarrow}$ in Definition 5.7, except that:

- states in $\mathscr{F}_{\mathscr{W}_0}^{\rightarrow}$ are additionally required to locally satisfy $\mathscr{D}$, and
- the transition function is additionally restricted so that transitions to a window $(\varrho, T, T, 0)$ are only allowed if there exists no window $(\varrho, H, T, 1)$ locally satisfying Π and $\mathscr{D}$ such that H and T coincide over $[\varrho^-, \varrho^+ - 1]$ and $M@\varrho^+ \in T \backslash H$ for some relational atom M.

Note that $\mathscr{F}_{\mathscr{W}_0}^{\rightarrow}$ is essentially deterministic since $\delta(\mathscr{W}, \sigma)$ contains at most one window for every state $\mathscr{W}$ and $\sigma \in \Sigma$. The following lemma provides the result analogous to Theorem 5.8 for the setting considered in this section.

Lemma 5.16
Program Π and dataset $\mathscr{D}$ have a stable model if and only if there exists an initial window $\mathscr{W}_0 = (\varrho_0, T_0, T_0, 0)$ locally satisfying Π and $\mathscr{D}$ with $\varrho_0 = [t_{\mathscr{D}}^{\min} - (t_{\Pi} + 1), t_{\mathscr{D}}^{\min} - 1]$, which mentions only constants and predicates from Π, and there exist words $\overleftarrow{w}$ and $\overrightarrow{w}$ over $2^{\mathsf{at}(\Pi, \mathscr{D})}$ such that all of the following conditions hold:

1. $\overleftarrow{w}$ and $\overrightarrow{w}$ are accepted by $\mathscr{B}_{\mathscr{W}_0}^{\leftarrow}$ and $\mathscr{F}_{\mathscr{W}_0}^{\rightarrow}$, respectively,
2. there is no initial window $\mathscr{W}_0' = (\varrho_0, H_0, T_0, b_0)$ locally satisfying Π and $\mathscr{D}$ such that $\overleftarrow{w}$ is accepted by $\mathscr{C}_{\mathscr{W}_0'}^{\leftarrow}$,
3. $\overleftarrow{w}$ mentions only constants and predicates from Π.

Proof
Assume that $\mathfrak{T}$ is a stable model of Π and $\mathscr{D}$. To construct the required $\mathscr{W}_0$, $\overleftarrow{w}$, and $\overrightarrow{w}$, we let $\ldots, \mathscr{W}_{-1}, \mathscr{W}_0, \mathscr{W}_1, \ldots$ be the $[t_{\mathscr{D}}^{\min} - (t_{\Pi} + 1), t_{\mathscr{D}}^{\min} - 1]$-decomposition of HT-interpretation $(\mathfrak{T}, \mathfrak{T})$, and we let $\mathscr{W}_i = (\varrho_i, H_i, T_i, b_i)$. Moreover, we let $\overleftarrow{w} = \sigma_{-1}\sigma_{-2}\cdots$ and $\overrightarrow{w} = \sigma_1\sigma_2\cdots$ be the words such that $\sigma_k = T_k \backslash T_{k+1}$ if $k < 0$, and $T_k \backslash T_{k-1}$ if $k > 0$. In what follows, we will show that the above defined $\mathscr{W}_0$, $\overleftarrow{w}$, and $\overrightarrow{w}$ satisfy the requirements from the lemma. First, we observe that $H_i = T_i$ and $b_i = 0$, for each $i \in \mathbb{Z}$. Furthermore, by Lemma 5.6, each $\mathscr{W}_i$ is a window locally satisfying Π and, since $\mathfrak{T}$ is a stable model of

Π and $\mathscr{D}$, we obtain that each $\mathscr{W}_i$ locally satisfies $\mathscr{D}$. Therefore, as required in the lemma, $\mathscr{W}_0$ is an initial window of the form $(\varrho_0, T_0, T_0, 0)$ with $\varrho_0 = [t_{\mathscr{D}}^{\min} - (t_\Pi + 1), t_{\mathscr{D}}^{\min} - 1]$, and it locally satisfies Π and $\mathscr{D}$. Thus, it remains to show that $\mathscr{W}_0$ mentions only constants and predicates from Π and that $\mathscr{W}_0$, $\overleftarrow{w}$, and $\overrightarrow{w}$ satisfy Conditions 1–3.

To show that Condition 1 holds, we observe that, by Lemma 5.6, $\mathscr{W}_0, \mathscr{W}_{-1}, \ldots$ is an accepting run of $\mathscr{A}_{\mathscr{W}_0}^{\leftarrow}$; moreover, since $H_i = T_i$ and $b_i = 0$ for all integers $i \leq 0$, this run is also accepting for $\mathscr{B}_{\mathscr{W}_0}^{\leftarrow}$. Analogously, $\mathscr{W}_0, \mathscr{W}_1, \ldots$ is an accepting run of $\mathscr{B}_{\mathscr{W}_0}^{\leftarrow}$. Now, suppose towards a contradiction that $\mathscr{F}_{\mathscr{W}_0}^{\rightarrow}$ does not accept $\overrightarrow{w}$. Then, by Definition 5.15, there exists a smallest integer $i > 0$ such that $\mathscr{W}_0, \ldots, \mathscr{W}_{i-1}$ is a run of $\mathscr{F}_{\mathscr{W}_0}^{\rightarrow}$ on $\sigma_1 \cdots \sigma_{i-1}$ (where this word is empty if $i = 1$), but $\mathscr{F}_{\mathscr{W}_0}^{\rightarrow}$ does not have a transition from $\mathscr{W}_{i-1}$ to $\mathscr{W}_i$ on σ_i. Next, we will show how to define windows $\mathscr{W}_i', \mathscr{W}_{i+1}', \ldots$ such that $\mathscr{W}_0, \ldots, \mathscr{W}_{i-1}, \mathscr{W}_i', \mathscr{W}_{i+1}', \ldots$ is an accepting run of $\mathscr{C}_{\mathscr{W}_0}^{\rightarrow}$ on $\overrightarrow{w}$, which will allow us to raise a contradiction.

Claim 5.17
There exist windows $\mathscr{W}_i', \mathscr{W}_{i+1}', \ldots$ locally satisfying $\mathscr{D}$ such that $\mathscr{W}_0, \ldots, \mathscr{W}_{i-1}, \mathscr{W}_i'$, $\mathscr{W}_{i+1}', \ldots$ is an accepting run of $\mathscr{C}_{\mathscr{W}_0}^{\rightarrow}$ on $\overrightarrow{w}$.

Proof of Claim 5.17
Recall that $\mathscr{W}_i = (\varrho_i, T_i, T_i, 0)$. Since $\mathscr{F}_{\mathscr{W}_0}^{\rightarrow}$ does not have a transition from $\mathscr{W}_{i-1}$ to $\mathscr{W}_i$ on σ_i, there exists a window $\mathscr{W}' = (\varrho_i, H', T_i, 1)$ locally satisfying Π and $\mathscr{D}$ such that H' coincides with T_i over $[\varrho_i^-, \varrho_{i-1}^+]$, and there exists an atom $M' \in \mathsf{at}(\Pi, \mathscr{D})$ satisfying $M'@\varrho_i^+ \in T_i \backslash H'$. We note also that since $\mathscr{W}'$ is a window and $H' \subseteq T_i$, there exists an interpretation $\mathfrak{H}'$ such that $\mathfrak{H}' \subseteq \mathfrak{T}$ and both items from Definition 5.2 hold. We will use $\mathfrak{H}'$ to define $\mathscr{W}_i', \mathscr{W}_{i+1}', \ldots$. To this end, we will say that an HT-interpretation $(\mathfrak{H}^*, \mathfrak{T}^*)$ *satisfies* a dataset $\mathscr{D}$ and program Π *over* an interval ϱ^* if and only if

- for every $M@\varrho \in \mathscr{D}$ and $t \in \varrho \cap \varrho^*$ we have $\mathfrak{H}^*, t \models M$, and
- $(\mathfrak{H}^*, \mathfrak{T}^*)$ satisfy the conditions of Definition 3.2 for each $t \in \varrho^*$.

We can use a construction similar to that in the proof of Theorem 3.4 to define the least interpretation $\mathfrak{H}$ such that $\mathfrak{H}' \subseteq \mathfrak{H} \subseteq \mathfrak{T}$, $\mathfrak{H}$ coincides with $\mathfrak{H}'$ over $(-\infty, \varrho_i^-)$, and $(\mathfrak{H}, \mathfrak{T})$ satisfies $\mathscr{D}$ and Π over $[\varrho_i^-, \infty)$. An important observation is that since Π is forward-propagating and $\mathscr{W}'$ locally satisfies Π and $\mathscr{D}$, interpretations $\mathfrak{H}'$ and $\mathfrak{H}$ also coincide over the interval ϱ_i.

Now, we let $\ldots, \mathscr{W}_{-1}', \mathscr{W}_0', \mathscr{W}_1', \ldots$, with $\mathscr{W}_j' = (\varrho_j', H_j', T_j', b_j')$, be the ϱ_0-decomposition of $(\mathfrak{H}, \mathfrak{T})$; hence, $\varrho_j' = \varrho_j$ and $T_j' = T_j$, for each j. We observe also that since $\mathfrak{H}$ coincides with $\mathfrak{H}'$ over $(-\infty, \varrho_i^+]$, H_i' agrees with H' on all relational atoms over $(-\infty, \varrho_i^+]$, so $M'@t \in T_i \backslash H_i'$, and hence $b_j' = 1$ for each $j \geq i$.

We will show that $\mathscr{W}_0, \ldots, \mathscr{W}_{i-1}, \mathscr{W}_i', \mathscr{W}_{i+1}', \ldots$ is an accepting run of $\mathscr{A}_{\mathscr{W}_0}^{\rightarrow}$ on $\overrightarrow{w}$ where all windows locally satisfy $\mathscr{D}$, and then, we will show that this run is also accepted by $\mathscr{C}_{\mathscr{W}_0}^{\rightarrow}$. To this end, we observe that since $\mathscr{W}_0, \ldots, \mathscr{W}_{i-1}$ is a run of $\mathscr{F}_{\mathscr{W}_0}^{\rightarrow}$ on $\sigma_1 \cdots \sigma_{i-1}$, it is also a run of $\mathscr{A}_{\mathscr{W}_0}^{\rightarrow}$ on $\sigma_1 \cdots \sigma_{i-1}$. Moreover, $\mathscr{W}_i'$ is an initial window, (since $H_i' \neq T_i$ and $b_i' = 1$) and, by an argument analogous to the proof of Lemma 5.6, each $\mathscr{W}_j'$ for $j \geq i$ locally satisfies Π and $\mathscr{D}$, and $\mathscr{W}_j', \mathscr{W}_{j+1}', \ldots$ is an accepting run of $\mathscr{A}_{\mathscr{W}_i'}^{\rightarrow}$ on $\sigma_{i+1}, \sigma_{i+2}, \ldots$. Hence, to show that $\mathscr{W}_0, \ldots, \mathscr{W}_{i-1}, \mathscr{W}_i', \mathscr{W}_{i+1}', \ldots$ is an accepting run of $\mathscr{A}_{\mathscr{W}_0}^{\rightarrow}$, it remains to show that $\mathscr{A}_{\mathscr{W}_0}^{\rightarrow}$ has a transition on σ_i from $\mathscr{W}_{i-1}$ to $\mathscr{W}_i'$. To do this,

we first show that H_i' coincides with H' over $[\varrho_i^-, \varrho_{i-1}^+]$. Indeed, consider an arbitrary fact $M@t$ with $M \in \mathsf{at}(\Pi, \mathscr{D})$ and $t \in [\varrho_i^-, \varrho_{i-1}^+]$; we will show that $M@t \in H_i'$ if and only if $M@t \in H'$. Note that since Π is forward-propagating, any atom $M \in \mathsf{at}(\Pi, \mathscr{D})$ is either of the form $\boxplus_{[0,\infty)} P(\mathbf{c})$ for some relational fact $P(\mathbf{c})$, or it does not mention future operators. Thus, if M does not mention future operators, the biconditional holds because $\mathfrak{H}$ and $\mathfrak{H}'$ coincide over the interval $(-\infty, \varrho_i^+]$. Now, suppose M is of the form $\boxplus_{[0,\infty)} P(\mathbf{c})$. If $\boxplus_{[0,\infty)} P(\mathbf{c})@t \in H'$, then $\boxplus_{[0,\infty)} P(\mathbf{c})@t \in H_i'$ since $H' \subseteq H_i'$ by construction of H_i'. Conversely, if $\boxplus_{[0,\infty)} P(\mathbf{c})@t \in H_i'$, we have that $\mathfrak{H}, t \models \boxplus_{[0,\infty)} P(\mathbf{c})$, and since $\mathfrak{H} \subseteq \mathfrak{T}$, we have $\mathfrak{T}, t \models \boxplus_{[0,\infty)} P(\mathbf{c})$, so $\boxplus_{[0,\infty)} P(\mathbf{c})@t \in T_i$. Then, since H' agrees with T_i over $[\varrho_i^-, \varrho_{i-1}^+]$, we conclude that $\boxplus_{[0,\infty)} P(\mathbf{c})@t \in H'$. Thus, the biconditional also holds in this case. With the above result, we can show that there is a transition in $\mathscr{A}_{\mathscr{W}_0}^{\rightarrow}$ from $\mathscr{W}_{i-1}$ to $\mathscr{W}_i'$; we can see this by checking that all the four conditions from Item 3 in Definition 5.4 hold. The first, third, and fourth conditions hold directly by the definition of $\mathscr{W}_i'$ and the fact that $\varrho_i' = \varrho_i$, $T_i' = T_i$, and $b_i' = 1$. The second condition states that H_i' and H_{i-1} must coincide over the interval $\varrho_{i-1} \cap \varrho_i$. To see this, recall that H_i' coincides with H' over this interval. In turn, H' coincides with T_i over this interval by definition, and T_i coincides with T_{i-1} over this interval by construction, but $T_{i-1} = H_{i-1}$, so H_i' coincides with H_{i-1} and the condition holds. Therefore, $\mathscr{W}_0, \ldots, \mathscr{W}_{i-1}, \mathscr{W}_i', \mathscr{W}_{i+1}', \ldots$ is an accepting run of $\mathscr{A}_{\mathscr{W}_0}^{\rightarrow}$ on $\overrightarrow{w}$ where all windows locally satisfy $\mathscr{D}$.

Furthermore, $\mathscr{W}_0, \ldots, \mathscr{W}_{i-1}, \mathscr{W}_i', \mathscr{W}_{i+1}', \ldots$ is also an accepting run of $\mathscr{C}_{\mathscr{W}_0}^{\rightarrow}$ on $\overrightarrow{w}$, since every window in the run after $\mathscr{W}_i'$ has 1 as its fourth component, and so the additional accepting condition of $\mathscr{C}_{\mathscr{W}_0}^{\rightarrow}$ is satisfied. This concludes the proof of Claim 5.17. □

Having proved the claim, we resume the proof of Lemma 5.16. Recall that $\mathscr{W}_0, \mathscr{W}_{-1}, \ldots$ is an accepting run of $\mathscr{A}_{\mathscr{W}_0}^{\leftarrow}$ on $\overleftarrow{w}$ and, as we showed in the claim above, $\mathscr{W}_0, \ldots, \mathscr{W}_{i-1}, \mathscr{W}_i', \mathscr{W}_{i+1}', \ldots$ is an accepting run of $\mathscr{C}_{\mathscr{W}_0}^{\rightarrow}$ on $\overrightarrow{w}$. Hence, as in the proof of Theorem 5.8, we can use these runs to construct an HT-model $(\mathfrak{H}'', \mathfrak{T})$ of Π and $\mathscr{D}$ such that $\mathfrak{H}'' \subsetneq \mathfrak{T}$; in particular, $\mathfrak{H}'' \models \mathscr{D}$, as ϱ_0^- is to the left of any integer mentioned in $\mathscr{D}$ and each window in the run $\mathscr{W}_0, \ldots, \mathscr{W}_{i-1}, \mathscr{W}_i', \mathscr{W}_{i+1}', \ldots$ locally satisfies $\mathscr{D}$. This, however, means that $\mathfrak{T}$ is not a stable model of Π and $\mathscr{D}$, which raises a contradiction.

To prove Condition 2, suppose towards a contradiction that there exists an initial window $\mathscr{W}_0' = (\varrho_0, H_0, T_0, b_0)$ which locally satisfies Π and $\mathscr{D}$, and such that there exits an accepting run $\mathscr{W}_0', \mathscr{W}_{-1}', \ldots$ of $\mathscr{C}_{\mathscr{W}_0'}^{\leftarrow}$ on $\overleftarrow{w}$. We observe that, by definition of $\overleftarrow{w}$, each $\mathscr{W}_i'$ is of the form $(\varrho_i, H_i, T_i, b_i)$, for some set H_i of metric atoms and $b_i \in \{0, 1\}$. We consider the least model $\mathfrak{H}'$ of all relational facts in $\bigcup_{i \leq 0} H_i$. The accepting conditions of $\mathscr{C}_{\mathscr{W}_0'}^{\leftarrow}$ ensure that there is an atom $P(\mathbf{c})$ and a time point $t \in (-\infty, \varrho_0^+]$ such that $\mathfrak{H}', t \not\models P(\mathbf{c})$ but $\mathfrak{T}, t \models P(\mathbf{c})$. By an argument analogous to that in the proof of Theorem 5.8, and using the facts that $\mathscr{W}_0'$ locally satisfies $\mathscr{D}$ and $\mathscr{W}_i'$ locally satisfies Π for each $i \leq 0$, we obtain that $(\mathfrak{H}', \mathfrak{T})$ satisfies $\mathscr{D}$ and Π over $(-\infty, \varrho_0^+]$. Then, using a construction similar to that in the proof of Theorem 3.4, we can extend $\mathfrak{H}'$ to the minimal interpretation $\mathfrak{H}$ such that $(\mathfrak{H}, \mathfrak{T})$ is an HT-model of Π and $\mathscr{D}$. Since Π is forward-propagating, and $(\mathfrak{H}', \mathfrak{T})$ already satisfies $\mathscr{D}$ and Π over $(-\infty, \varrho_0^+]$, we have that $\mathfrak{H}$ and $\mathfrak{H}'$ agree over $(-\infty, \varrho_0^+]$. But then, $\mathfrak{H} \subsetneq \mathfrak{T}$ since, as we have shown, there is $t \in (-\infty, \varrho_0^+]$ such that $\mathfrak{H}', t \not\models P(\mathbf{c})$ but $\mathfrak{T}, t \models P(\mathbf{c})$. Thus, $\mathfrak{T}$ is not stable, which raises a contradiction.

To prove Condition 3, we define a sequence $\mathfrak{H}_0, \mathfrak{H}_1, \ldots$ of interpretations as follows:

– $\mathfrak{H}_0$ is the least model of $\mathscr{D}$,
– $\mathfrak{H}_\alpha$, for a successor ordinal α, is the least interpretation such that for each rule of Form (1) in $\mathsf{ground}(\Pi)$, and for each time point t, if $\mathfrak{H}_{\alpha-1}, t \models M_i$ for each $1 \leq i \leq k$ and $\mathfrak{T}, t \not\models M_j$ for each $k+1 \leq j \leq m$, then $\mathfrak{H}_\alpha, t \models M$,
– $\mathfrak{H}_\alpha$, for a limit ordinal α, is $\bigcup_{\beta < \alpha} \mathfrak{H}_\beta$.

By the proof of Theorem 3.4, each $\mathfrak{H}_\alpha$ is well defined and $\mathfrak{T} = \mathfrak{H}_{\omega_1}$. Hence, to prove Condition 3, it suffices to show by transfinite induction that, for every ordinal α, if $\mathfrak{H}_\alpha, t \models P(\mathbf{c})$ for some relational atom $P(\mathbf{c})$ and $t < t_{\mathscr{D}}^{\min}$, then P and all constants in $\mathbf{c}$ occur in Π. In the base case $\mathfrak{H}_0$ is the least model of $\mathscr{D}$. Since $\mathscr{D}$ is bounded, all the facts it mentions are over intervals contained in $[t_{\mathscr{D}}^{\min}, \infty)$, and so, the statement holds. In the inductive step for a successor ordinal α, we suppose towards a contradiction that $\mathfrak{H}_\alpha, t \models P(\mathbf{c})$ and $\mathfrak{H}_{\alpha-1}, t \not\models P(\mathbf{c})$, for some relational atom $P(\mathbf{c})$ and $t < t_{\mathscr{D}}^{\min}$, such that P or some constant in $\mathbf{c}$ does not occur in Π. Hence, there exists a rule r in $\mathsf{ground}(\Pi, \mathscr{D})$ whose head is $P(\mathbf{c})$, whose positive body atoms are satisfied by $\mathfrak{H}_{\alpha-1}$ at t, and whose negated body atoms are satisfied by $\mathfrak{T}$ at t. Thus, P appears in Π, so there exists a constant c in $\mathbf{c}$ which does not appear in Π. By the safety and the normal form of r, the constant c needs to occur in a relational atom M such that either M or $\boxminus_\varrho M$, or $M' \mathscr{S}_\varrho M$ is a positive body atom in r. Each of these cases, however, implies that $\mathfrak{H}_{\alpha-1}, t' \models M$, for some $t' \leq t$, which violates the induction hypothesis. In the inductive step for a limit ordinal α, we have $\mathfrak{H}_\alpha = \bigcup_{\beta < \alpha} \mathfrak{H}_\beta$; since the claim holds for each $\mathfrak{H}_\beta$ by induction hypothesis, it holds also for $\mathfrak{H}_\alpha$.

We observe that the above result not only shows that Condition 3 holds, but also that $\mathscr{W}_0$ mentions only constants and predicates from Π, which completes the proof of the first implication from the lemma.

To show the reverse implication from Lemma 5.16, let $\mathscr{W}_0 = (\varrho_0, T_0, T_0, 0)$, $\overleftarrow{w}$, and $\overrightarrow{w}$ be as described in Lemma 5.16. In particular, by Condition 2, $\mathscr{B}_{\mathscr{W}_0}^{\leftarrow}$ has an accepting run $\mathscr{W}_0, \mathscr{W}_{-1}, \ldots$ on $\overleftarrow{w}$ and $\mathscr{F}_{\mathscr{W}_0}^{\rightarrow}$ has an accepting run $\mathscr{W}_0, \mathscr{W}_1, \ldots$ on $\overrightarrow{w}$, where we let $\mathscr{W}_i = (\varrho_i, T_i, T_i, 0)$. We will show that the least model $\mathfrak{T}$ of relational facts in $\bigcup_{i \in \mathbb{Z}} T_i$ is a stable model of Π and $\mathscr{D}$. Using an argument analogous to the proof of Theorem 5.8, we obtain that $(\mathfrak{T}, \mathfrak{T})$ is an HT-model of Π and $\mathscr{D}$. Now, suppose towards a contradiction that $\mathfrak{T}$ is not a stable model, so there is an interpretation $\mathfrak{H} \subsetneq \mathfrak{T}$ such that $(\mathfrak{H}, \mathfrak{T})$ is an HT-model of Π and $\mathscr{D}$. We let $\ldots, \mathscr{W}_{-1}', \mathscr{W}_0', \mathscr{W}_1', \ldots$ be the $[t_{\mathscr{D}}^{\min} - (t_\Pi + 1), t_{\mathscr{D}}^{\min} - 1]$-decomposition of $(\mathfrak{H}, \mathfrak{T})$, where we let $\mathscr{W}_i' = (\varrho_i', H_i', T_i', b_i')$. By construction, $\varrho_i' = \varrho_i$ and $T_i' = T_i$. Moreover, by Definition 5.5, $\mathscr{W}_0'$ is an initial window, and it is straightforward to verify that $\mathscr{W}_i'$ locally satisfies Π and $\mathscr{D}$, for each $i \in \mathbb{Z}$. Now, by Lemma 5.6, $\mathscr{W}_0', \mathscr{W}_{-1}', \ldots$ is an accepting run of $\mathscr{A}_{\mathscr{W}_0'}^{\leftarrow}$ on $\overleftarrow{w}$, and $\mathscr{W}_0', \mathscr{W}_1', \ldots$ is an accepting run of $\mathscr{A}_{\mathscr{W}_0'}^{\rightarrow}$ on $\overrightarrow{w}$. Since $\mathfrak{H} \subsetneq \mathfrak{T}$, there exists $i \in \mathbb{Z}$ such that $\mathfrak{T}, \varrho_i^+ \models M$ but $\mathfrak{H}, \varrho_i^+ \not\models M$ for some relational atom $M \in \mathsf{at}(\Pi, \mathscr{D})$, which implies $H_i \neq T_i$. If $i \leq 0$, then $b_j' = 1$ for all $j \leq i$, and so $\mathscr{C}_{\mathscr{W}_0'}^{\leftarrow}$ accepts $\overleftarrow{w}$, which contradicts Condition 2. Otherwise, let i be the least (positive) integer such that $\mathfrak{T}, \varrho_i^+ \models M$ but $\mathfrak{H}, \varrho_i^+ \not\models M$, for some relational atom $M \in \mathsf{at}(\Pi, \mathscr{D})$. Observe that this implies $H_j' = T_j$ for each $j < i$, and $M@\varrho_i^+ \in T_i \backslash H_i'$. By construction, H_i' coincides with H_{i-1}' over $[\varrho_i^-, \varrho_i^+ - 1]$; however, since $H_{i-1}' = T_{i-1}$, and T_{i-1} coincides with T_i over $[\varrho_i^-, \varrho_i^+ - 1]$, we have that H_i' coincides with T_i over this interval. To sum

up, recall that $\mathscr{W}_i$ is of the form $(\varrho_i, T_i, T_i, 0)$, whereas, by construction, $\mathscr{W}_i'$ is of the form $(\varrho_i, H_i', T_i, 1)$; furthermore, H_i' coincides with T_i over $[\varrho_i^-, \varrho_i^+ - 1]$, and there exists a relational atom M with $M@\varrho^+ \in T_i \backslash H_i'$. This, by Definition 5.15, implies that the automaton $\mathscr{F}_{\mathscr{W}_0}^{\rightarrow}$ cannot have a transition to $\mathscr{W}_i$, and so, $\mathscr{W}_0, \mathscr{W}_1, \ldots$ is not an accepting run of $\mathscr{F}_{\mathscr{W}_0}^{\rightarrow}$ on $\overrightarrow{w}$, which raises a contradiction. □

Next, we use Lemma 5.16 to establish a tight PSPACE bound for reasoning in DatalogMTL$_{\mathsf{FP}}^{\neg}$.

Theorem 5.18
Checking whether a DatalogMTL$_{\mathsf{FP}}^{\neg}$ program and a bounded dataset have a stable model is PSPACE-complete with respect to data complexity.

Proof
For the lower bound, we observe that Wałęga *et al.* (2020a) showed PSPACE-hardness in data complexity of checking existence of models for a class of programs which is strictly smaller than the class of positive DatalogMTL$_{\mathsf{FP}}^{\neg}$ programs. Their reduction can be modified in a straightforward way so that the involved dataset is bounded. Then, Theorem 3.7 directly implies that the same lower bound holds for all (i.e., not necessarily positive) DatalogMTL$_{\mathsf{FP}}^{\neg}$ programs, as required.

For the upper bound, by Lemma 5.16, it suffices to show that checking existence of a window $\mathscr{W}_0$ and words $\overleftarrow{w}$ and $\overrightarrow{w}$ satisfying the properties described in the statement of the lemma is feasible in PSPACE. First, we observe that $\mathscr{W}_0$ is over $\varrho_0 = [t_{\mathscr{D}}^{\min} - (t_{\Pi} + 1), t_{\mathscr{D}}^{\min} - 1]$, so its length does not depend on $\mathscr{D}$. Hence, $\mathscr{W}_0$ is polynomially large (in the size of the representation of $\mathscr{D}$), and so it can be guessed in PSPACE; moreover, one can check in PSPACE whether $\mathscr{W}_0$ locally satisfies Π and $\mathscr{D}$, and whether it mentions only constants and predicates from Π. Next, we show how to verify existence of words $\overleftarrow{w}$ and $\overrightarrow{w}$ over $2^{\mathsf{at}(\Pi, \mathscr{D})}$ such that $\mathscr{W}_0, \overleftarrow{w}$, and $\overrightarrow{w}$ satisfy Conditions 1–3. To verify existence of a word $\overleftarrow{w}$ accepted by $\mathscr{B}_{\mathscr{W}_0}^{\leftarrow}$ (first part of Condition 1) which is not accepted by any $\mathscr{C}_{\mathscr{W}_0'}^{\leftarrow}$ (Condition 2) we can use the approach from the proof of Theorem 5.14. We observe that $\mathscr{W}_0$ mentions only constants and predicates from Π, and so, the same holds for windows $\mathscr{W}_0'$ from Condition 2. Moreover, by Condition 3, $\overleftarrow{w}$ also mentions only constants and predicates from Π, and so the above check can be performed independently of $\mathscr{D}$.

It remains to be shown that checking existence of a word $\overrightarrow{w}$ accepted by $\mathscr{F}_{\mathscr{W}_0}^{\rightarrow}$ (that is, the second part of Condition 1) is feasible in PSPACE. To this end, we check existence of an accepting run $\mathscr{W}_0, \mathscr{W}_1, \ldots$ of $\mathscr{F}_{\mathscr{W}_0}^{\rightarrow}$ in two steps. First, we guess windows $\mathscr{W}_1, \ldots, \mathscr{W}_j$ one by one, where $j = t_{\mathscr{D}}^{\max} - t_{\mathscr{D}}^{\min} + 2t_{\Pi} + 2$, and second, we check if $\mathscr{F}_{\mathscr{W}_j}^{\rightarrow}$ has an accepting run. We observe that each of the windows $\mathscr{W}_1, \ldots, \mathscr{W}_j$ is of polynomial size, and so guessing them one by one, as well as checking that $\mathscr{F}_{\mathscr{W}_0}^{\rightarrow}$ has transitions between consecutive windows, is feasible in PSPACE. To check non-emptiness of the language of $\mathscr{F}_{\mathscr{W}_j}^{\rightarrow}$, we construct for it an automaton $\widetilde{\mathscr{F}_{\mathscr{W}_j}^{\rightarrow}}$ in a similar way as we constructed $\widetilde{X_{\mathscr{W}_j}^{\rightarrow}}$ for $X_{\mathscr{W}_j}^{\rightarrow}$ in the proof of Theorem 5.14. The difference, however, is that the set of states of $\widetilde{\mathscr{F}_{\mathscr{W}_j}^{\rightarrow}}$ is the quotient set of $\sim$ between only those states of $\mathscr{F}_{\mathscr{W}_j}^{\rightarrow}$ whose first elements are intervals located entirely to the right of $t_{\mathscr{D}}^{\max} + t_{\Pi}$. For any such state $\mathscr{W}_i = (\varrho_i, T_i, T_i, 0)$, checking whether it locally satisfies $\mathscr{D}$ simply amounts to verifying that $M@t \in T_i$ for each

$M \in \mathsf{at}(\Pi, \mathscr{D})$ such that $M@\varrho_j^- \in T_j$, and each $t \in \varrho_i$. Indeed, this follows from two observations: first, for any atom $M \in \mathsf{at}(\Pi, \mathscr{D})$ and two arbitrary time points $t, t' > t_{\mathscr{D}}^{\max} + t_\Pi$, $\mathscr{D} \models M@t$ if and only if $\mathscr{D} \models M@t'$; second, $\varrho_j^- > t_{\mathscr{D}}^{\max} + t_\Pi$, by definition of j. Then, using an argument analogous to the proof of Lemma 5.10, we can show that $\overrightarrow{\mathscr{F}_{\mathscr{W}_j}}$ and $\widetilde{\overrightarrow{\mathscr{F}_{\mathscr{W}_j}}}$ are equivalent. Hence, it remains to check if the language of the latter automaton is non-empty. By construction, each state of the automaton $\widetilde{\overrightarrow{\mathscr{F}_{\mathscr{W}_j}}}$ can be represented in polynomial space, so the non-emptiness check is feasible in PSPACE using a standard on-the-fly approach.   □

The assumption that $\mathscr{D}$ is bounded has been used to ensure existence of a time point such that no fact of $\mathscr{D}$ holds to the left of it. Thus, our results can be extended to show that reasoning is still PSPACE in data complexity for datasets where intervals are only bounded on the left. Furthermore, none of our results in this section depend on the direction of time. Indeed, we can define the *backward-propagating* fragment of DatalogMTL¬ (analogously to DatalogMTL¬FP) as the set of programs where operators $\diamondsuit$, $\boxminus$, and $\mathscr{S}$ are disallowed in rule bodies, and operator $\boxplus$ is disallowed in the head. Then, we can obtain an analogous set of results for the backward-propagating fragment and show that reasoning in such fragment is also PSPACE in data complexity.

6 Related work

Positive DatalogMTL (Brandt *et al.* 2018) has been studied over both the rational (Brandt *et al.* 2018) and the integer (Wałęga *et al.* 2020a) timelines. In both cases, the main reasoning tasks are ExpSpace-complete for combined complexity (Brandt *et al.* 2018) and PSPACE-complete for data complexity (Wałęga *et al.* 2019). Low complexity fragments (Wałęga *et al.* 2020b; 2021) and alternative semantics (Ryzhikov *et al.* 2019) have also been studied. Practical reasoning algorithms for positive DatalogMTL have been recently proposed and implemented in the MeTeoR system (Wang *et al.* 2022).

DatalogMTL¬ is an extension of DatalogMTL; therefore, it also extends other prominent temporal rule languages captured by DatalogMTL such as Datalog$_{1S}$ (Chomicki and Imieliński 1988; 1989) and Templog (Abadi and Manna 1989). In turn, DatalogMTL¬FP generalises the forward-propagating fragment of DatalogMTL introduced by Wałęga *et al.* (2019), and it is thus related to other forward-propagating temporal logics proposed in the literature (Baldor and Niu 2012; Ronca *et al.* 2018; Basin *et al.* 2018). Our stable model semantics (Wałęga *et al.* 2021) extends the semantics for stratified DatalogMTL¬ programs (Tena Cucala *et al.* 2021).

Our approach is closely related to a recently proposed family of non-monotonic temporal logics which simultaneously support ASP and modal temporal operators. TEL (Cabalar and Vega 2007; Aguado *et al.* 2013; Cabalar *et al.* 2018) combines propositional ASP with operators from linear temporal logic (Pnueli 1977). Metric equilibrium logic (MEL) (Cabalar *et al.* 2020) extends TEL with metric temporal operators that are roughly equivalent to our past operators $\mathscr{S}_{[0,k]}$ and $\boxminus_k$, and their future counterparts. Both logics introduce non-monotonic semantics for negation based on stable models, which are defined – as in our work – analogously to the models of equilibrium logic. However, TEL and MEL differ from our approach. First, they allow formulas supporting all Boolean connectives; therefore, they can represent disjunction between propositions,

as well as 'existential' formulas using diamond operators; in contrast, in DatalogMTL¬ the use of logical connectives and temporal operators is restricted so that all formulas are shaped as rules similar in spirit to those of Datalog (see Definition 3.1). Second, they are propositional logics, so they do not allow universally quantified variables. Finally, the semantics of TEL and MEL are defined on integer timelines with a least time point, whereas we consider both the full integer timeline as well as the dense rational timeline. In terms of complexity, checking whether a formula has a stable model is known to be PSpace-hard for both TEL and MEL, and feasible in ExpSpace for TEL.

Our approach is also related to the LARS language (Beck *et al.* 2018) for stream reasoning. LARS is also a rule-based language allowing for negation interpreted under stable model semantics. The differences between DatalogMTL¬ and LARS are as follows. First, conjuncts in the body of LARS rules are formulas constructed using temporal operators and unrestricted combinations of Boolean connectives; in contrast, body conjuncts in DatalogMTL¬ are metric atoms, which do not mention Boolean operators. Second, LARS does not allow for metric operators (only LTL-style boxes and diamonds), but it allows for window operators that have no counterpart in DatalogMTL¬. Third, LARS rules are meant to be interpreted at individual time points, so the notion of a stable model in LARS is always relative to a time point (e.g., a LARS interpretation can be a stable model of a program at t, but not at $t + 1$); in contrast, a stable model of a DatalogMTL¬ program satisfies each rule at *every* time point in the timeline (see Definition 3.2). Finally, LARS interpretations are defined over bounded intervals of the integer timeline, whereas DatalogMTL¬ interpretations are defined over the full integer or rational timeline. Checking whether a LARS program and dataset have a stable model at a time point t is PSpace-complete.

The use of ASP for temporal reasoning has been intensively explored, especially in the context of stream reasoning. Streamlog (Zaniolo 2012) introduces a variant of Datalog with negation where atoms are "time-stamped," in the sense that the first term of an atom is a natural number representing the time point where the atom holds. Rules in Streamlog must be forward-propagating in time; furthermore, programs must be locally stratified, which ensures that they have a unique stable model. Do *et al.* (2011) present an approach which combines a general-purpose ASP solver with a monotonic stream reasoner to support ASP in stream reasoning; in this approach, both systems are treated as black boxes. Another approach that supports ASP on a temporal setting is *time-decaying reasoning* (Gebser *et al.* 2013), which relies on programs similar to Datalog, but where each fact and rule need to hold only over a fixed time interval; this behaviour is used to capture changing information from a sliding window over a sequence of temporal data. More recently, negation-free DatalogMTL¬ has been applied in the context of monotonic stream reasoning (Wałęga *et al.* 2019), and we see our current work as providing the foundations for extending this approach with support for ASP.

7 Conclusion and future work

We extended DatalogMTL with negation-as-failure under stable model semantics and shown that reasoning in this language is undecidable over the rational timeline but ExpSpace in data complexity over the integers. We also studied the forward-propagating fragment and shown that, although reasoning remains undecidable over the

rational timeline, it is becomes PSpace-complete in data complexity over the integers (thus no harder than in the negation-free case).

We see many avenues for future work. The more immediate challenge is to provide tight data complexity bounds for reasoning in the full language over the integer timeline, where we currently have an ExpSpace upper bound and a lower bound for data complexity. We also plan to consider combined complexity and identify fragments of the language where reasoning becomes decidable over the rational timeline. Finally, we are planning to develop practical algorithms and implement them as an extension the MeTeoR reasoner, that is currently allows for reasoning with positive DatalogMTL programs only (Wang *et al.* 2022).

References

ABADI, M. AND MANNA, Z. 1989. Temporal logic programming. *Journal of Symbolic Computation 8,* 3, 277–295.

ABITEBOUL, S., HULL, R. AND VIANU, V. 1995. *Foundations of Databases.* Addison-Wesley.

AGUADO, F., CABALAR, P., DIÉGUEZ, M., PÉREZ, G. AND VIDAL, C. 2013. Temporal equilibrium logic: A survey. *Journal of Applied Non-Classical Logics 23,* 1–2, 2–24.

BAIER, C. AND KATOEN, J. 2008. *Principles of Model Checking.* MIT Press.

BALDOR, K. AND NIU, J. 2012. Monitoring dense-time, continuous-semantics, metric temporal logic. In *Runtime Verification, Third International Conference, Revised Selected Papers*, vol. 7687, 245–259.

BASIN, D., KLAEDTKE, F. AND ZĂLINESCU, E. 2018. Algorithms for monitoring real-time properties. *Acta Informatica 55,* 4, 309–338.

BECK, H., DAO-TRAN, M. AND EITER, T. 2018. LARS: A logic-based framework for analytic reasoning over streams. *Artificial Intelligence 261*, 16–70.

BOZZELLI, L. AND PEARCE, D. 2015. On the complexity of temporal equilibrium logic. In *Proceedings of the 30th Annual ACM/IEEE Symposium on Logic in Computer Science*, 645–656.

BRANDT, S., KALAYCI, E. G., RYZHIKOV, V., XIAO, G. AND ZAKHARYASCHEV, M. 2018. Querying log data with metric temporal logic. *Journal of Artificial Intelligence Resesearch 62*, 829–877.

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 *Proceedings of the 31st AAAI Conference on Artificial Intelligence*, 1070–1076.

BROOKS, D. R., ERDEM, E., ERDOGAN, S. T., MINETT, J. W. AND RINGE, D. 2007. Inferring phylogenetic trees using answer set programming. *Journal of Automated Reasoning 39,* 4, 471–511.

BRZOSKA, C. 1998. Programming in metric temporal logic. *Theoretical Computer Science 202,* 1–2, 55–125.

CABALAR, P. AND DEMRI, S. 2011. Automata-based computation of temporal equilibrium models. In *International Symposium on Logic-Based Program Synthesis and Transformation, Revised Selected Papers*, 57–72.

CABALAR, P., DIEGUEZ, M., SCHAUB, T. AND SCHUHMANN, A. 2020. Towards metric temporal answer set programming. *Theory and Practice of Logic Programming 20,* 5, 783–798.

CABALAR, P., KAMINSKI, R., SCHAUB, T. AND SCHUHMANN, A. 2018. Temporal answer set programming on finite traces. *Theory and Practice of Logic Programming 18,* 3–4, 406–420.

CABALAR, P. AND VEGA, G. P. 2007. Temporal equilibrium logic: A first approach. In *Proceedings of the 11th International Conference on Computer Aided Systems Theory*, 241–248.

CHOMICKI, J. AND IMIELIŃSKI, T. 1988. Temporal deductive databases and infinite objects. In *Proceedings of thes 7th ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems*, 61–73.

CHOMICKI, J. AND IMIELIŃSKI, T. 1989. Relational specifications of infinite query answers. In *Proceedings of the 1989 ACM SIGMOD International Conference on Management of Data*, 174–183.

DO, T. M., LOKE, S. W. AND LIU, F. 2011. Answer set programming for stream reasoning. In *Proceedings of the 24th Canadian Conference on Artificial Intelligence*, vol. 6657. Springer, 104–109.

EITER, T., IANNI, G. AND KRENNWALLNER, T. 2009. Answer set programming: A primer. In *Reasoning Web International Summer School*. Springer, 40–110.

FABER, W., LEONE, N. AND PFEIFER, G. 2004. Recursive aggregates in disjunctive logic programs: Semantics and complexity. In *Proceedings of the 9th European Conference on Logics in Artificial Intelligence*, 200–212.

GEBSER, M., GROTE, T., KAMINSKI, R., OBERMEIER, P., SABUNCU, O. AND SCHAUB, T. 2013. Answer set programming for stream reasoning. CoRR arXiv:1301.1392 [cs.AI].

GELFOND, M. AND LIFSCHITZ, V. 1988. The stable model semantics for logic programming. In *Logic Programming, Proceedings of the Fifth International Conference and Symposium*, 1070–1080.

HEYTING, A. 1930. Die formalen regeln der intuitionistischen logik. In *Sitzungsberichte der Preussischen Akademie der Wissenschaften, physikalisch-mathematische klass*, 42–56.

KOYMANS, R. 1990. Specifying real-time properties with metric temporal logic. *Real-Time Systems 2,* 4, 255–299.

NOGUEIRA, M., BALDUCCINI, M., GELFOND, M., WATSON, R. AND BARRY, M. 2001. An A-Prolog decision support system for the space shuttle. In *Proceedings of the 1st International Answer Set Programming Workshop*.

PEARCE, D. 1996. A new logical characterisation of stable models and answer sets. In *Proceedings of the International Workshop on Non-monotonic Extensions of Logic Programming*, 57–70.

PNUELI, A. 1977. The temporal logic of programs. In *Proceedings of the 18th Annual Symposium on Foundations of Computer Science*, 46–57.

RONCA, A., KAMINSKI, M., CUENCA GRAU, B. AND HORROCKS, I. 2018. The window validity problem in rule-based stream reasoning. In *Proceedings of the Sixteenth International Conference on Principles of Knowledge Representation and Reasoning*, 571–580.

RONCA, A., KAMINSKI, M., CUENCA GRAU, B., MOTIK, B. AND HORROCKS, I. 2018. Stream reasoning in temporal Datalog. In *Proceedings of the 32nd AAAI Conference on Artificial Intelligence*, 1941–1948.

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 *Proceedings of the 28th International Joint Conference on Artificial Intelligence*, 1851–1857.

TENA CUCALA, D., WAŁĘGA, P. A., CUENCA GRAU, B. AND KOSTYLEV, E. V. 2021. Stratified negation in Datalog with metric temporal operators. In *Proceedings of the 35th AAAI Conference on Artificial Intelligence*, 6488–6495.

WANG, D., HU, P., WAŁĘGA, P. A. AND CUENCA GRAU, B. 2022. MeTeoR: Practical reasoning in Datalog with metric temporal operators. In *Proceedings of the 36th AAAI Conference on Artificial Intelligence*.

WAŁĘGA, P., CUENCA GRAU, B. AND KAMINSKI, M. 2019. Reasoning over streaming data in metric temporal Datalog. In *Proceedings of the 33rd AAAI Conference on Artificial Intelligence*, 1941–1948.

WAŁĘGA, P., CUENCA GRAU, B., KAMINSKI, M. AND KOSTYLEV, E. V. 2020a. DatalogMTL over the integer timeline. In *Proceedings of the 17th International Conference on the Principles of Knowledge Representation and Reasoning*, 526–541.

WAŁĘGA, P. A., CUENCA GRAU, B., KAMINSKI, M. AND KOSTYLEV, E. V. 2019. DatalogMTL: Computational complexity and expressive power. In *Proceedings of the 28th International Joint Conference on Artificial Intelligence*, 1886–1892.

WAŁĘGA, P. A., CUENCA GRAU, B., KAMINSKI, M. AND KOSTYLEV, E. V. 2020b. Tractable fragments of Datalog with metric temporal operators. In *Proceedings of the 29th International Joint Conference on Artificial Intelligence*, 1919–1925.

WAŁĘGA, P. A., TENA CUCALA, D. J., KOSTYLEV, E. V. AND CUENCA GRAU, B. 2021. DatalogMTL with negation under stable models semantics. In *Proceedings of the 18th International Conference on Principles of Knowledge Representation and Reasoning*, 609–618.

WAŁĘGA, P. A., ZAWIDZKI, M. AND CUENCA GRAU, B. 2021. Finitely materialisable Datalog programs with metric temporal operators. In *Proceedings of the 18th International Conference on Principles of Knowledge Representation and Reasoning*, 619–628.

ZANIOLO, C. 2012. Logical foundations of continuous query languages for data streams. In *Proceedings of the 2nd International Workshop on Datalog in Academia and Industry*, 177–189.