*TPLP*: Page 1–66. © The Author(s), 2026. Published by Cambridge University Press. This is an Open Access article, distributed under the terms of the Creative Commons Attribution licence (https://creativecommons.org/licenses/by/4.0/), which permits unrestricted re-use, distribution and reproduction, provided the original article is properly cited.

doi:10.1017/S1471068426100714

# *Implementing Metric Temporal Answer Set Programming*[*]

SUSANA HAHN and ARVID BECKER
*Institute of Computer Science, University of Potsdam, Germany*
(*e-mails:* hahnmartinlu@uni-potsdam.de, arvidbecker@uni-potsdam.de)

PEDRO CABALAR
*Computer Science, University of Corunna, Spain*
(*e-mail:* cabalar@udc.es)

MARTÍN DIÉGUEZ
*Computer Science, Université d'Angers, France*
(*e-mail:* martin.dieguezlodeiro@univ-angers.fr)

JAVIER ROMERO and TORSTEN SCHAUB
*Department of Computer Science, University of Potsdam, Germany*
(*e-mails:* javier@cs.uni-potsdam.de, torsten@cs.uni-potsdam.de)

*submitted 25 March 2025 UTC; revised 25 March 2026 UTC; accepted 24 July 2026 UTC*

**Abstract**

We develop a computational approach to Metric Answer Set Programming (ASP) to express quantitative temporal constrains, such as durations and deadlines. We investigate two specific fragments: plain metric logic programs, restricted to local temporal constraints, and general metric logic programs, which allow for arbitrary metric formulas. A central challenge in this context is maintaining scalability when dealing with fine-grained timing constraints, which can significantly exacerbate grounding bottleneck of ASP. To address this issue, we propose translations of both fragments into standard ASP and ASP extended with difference constraints, a simplified form of linear constraints, and prove their correctness and completeness. Our implementation, realized via meta-encodings, effectively decouples metric ASP from the granularity of time, resulting in a solution that is independent of time precision.

*KEYWORDS:* metric answer set programming, equilibrium logic, difference constraints

[*] Research partially funded by DFG grant SCHA 550/15, Germany.

Downloaded from https://www.cambridge.org/core. UNIVERSIDAD AUTONOMA METROPOLITANA, on 08 Nov 2026 at 14:47:57, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://doi.org/10.1017/S1471068426100714

## 1 Introduction

Metric temporal logics (Fisher *et al.* 2005) allow for expressing quantitative temporal constrains, like durations and deadlines. As an example, consider the dentist scenario (Mellarkod 2007): *"Ram is at his office and has a dentist appointment in one hour. For the appointment, he needs his insurance card which is at home and cash to pay the doctor. He can get cash from the nearby Atm. Table 1 shows the time in minutes needed to travel between locations: Dentist, Home, Office and Atm. For example, the time needed to travel between Ram's office to the Atm is 20 minutes. The available actions are: moving from one location to another and picking items such as cash or insurance. The goal is to find a plan which takes Ram to the doctor on time."* This example combines planning and scheduling, and nicely illustrates the necessity to combine qualitative and quantitative temporal constraints.

Extensions to the Logic of Here-and-There and Equilibrium Logic (Pearce 1997) were developed in Cabalar *et al.* (2020b); Becker *et al.* (2024) to semantically ground the incorporation of metric constraints into Answer Set Programming (ASP; Lifschitz 2019).[1] Building upon these semantic foundations, we develop a computational approach to metric ASP. A central challenge in this is to maintain scalability when dealing with fine-grained timing constraints, which can significantly exacerbate ASP's grounding bottleneck. To address this issue, we leverage extensions of ASP with difference constraints (Janhunen *et al.* 2017), a simplified form of linear constraints, to handle time-related aspects externally. This approach effectively decouples metric ASP from the granularity of time, resulting in a solution that is unaffected by time precision. In detail, we (i) propose translations of metric logic programs into regular logic programs and their extension with difference constraints, (ii) prove the completeness and correctness of both translations in terms of equilibrium logic and its extensions with metric and difference constraints, (iii) describe an implementation of both translations in terms of meta-encodings, and (iv) present empirical performance results with novel metric encodings across three different domains, namely, the dentist scenario presented above, multi-agent synchronization, and job scheduling. From a modeling perspective, we may consider metric logic programs as a high-level modeling language for logic programs with difference constraints in the context of temporal domains. Our meta-programming approach enables the rapid exploration of alternative logical designs without requiring modifications to an underlying solver, such as the temporal ASP system *telingo* (Cabalar *et al.* 2020a). Instead, our approach provides a versatile environment suited for both theoretical research and the *generation* of system blueprints.

We begin by examining a restricted, yet expressive, fragment of our language, which allows the next operator to be constrained by a temporal interval. Subsequently, we extend this approach to encompass the full language. First, we demonstrate how arbitrary metric formulas can be transformed into a logic programming representation. Then, we

[1] While Cabalar *et al.* (2020b) equate time steps with state indices in a linear temporal model, Becker *et al.* (2024) pursue a more general approach by associating each state with an arbitrary natural number representing a specific point in time.

Downloaded from https://www.cambridge.org/core. UNIVERSIDAD AUTONOMA METROPOLITANA, on 08 Nov 2026 at 14:47:57, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://doi.org/10.1017/S1471068426100714

Table 1. *Durations between locations*

| | *Dentist* | *Home* | *Office* | *Atm* |
|---|---|---|---|---|
| *Dentist* | 0 | 20 | 30 | 40 |
| *Home* | 20 | 0 | 15 | 15 |
| *Office* | 30 | 15 | 0 | 20 |
| *Atm* | 40 | 15 | 20 | 0 |

illustrate how the techniques developed for the restricted fragment can be generalized to accommodate this more comprehensive format.

### *Related work*

Pioneering efforts to extend ASP with linear integer constraints for expressing quantitative temporal relations were introduced by Baselice *et al.* (2005) and Mellarkod (2007). These early approaches ultimately inspired the development of modern hybrid ASP systems, such as *clingcon* (Banbara *et al.* 2017), *clingo*[DL] (Janhunen *et al.* 2017), and *ezcsp* (Balduccini 2009). For instance, the latter was used by Balduccini *et al.* (2017) to address planning problems involving mixed discrete-continuous dynamics. While these systems effectively manage numerical constraints, they lack any support for high-level temporal languages and reasoning.

The logical foundation of our work lies in Metric Equilibrium Logic, as defined by Becker *et al.* (2024) as an extension of Linear temporal equilibrium logic (Aguado *et al.* 2023). The latter serves as the basis for the temporal ASP system *telingo* (Cabalar *et al*. 2020a), which supports qualitative temporal reasoning but cannot express metric constraints over time intervals. We bridge this gap by providing a general definition of metric logic programs and corresponding implementations based on meta-encodings, which may ultimately serve as blueprints for a future metric extension of *telingo*.

Metric concepts have also been explored in stream reasoning, most notably in the *lars* framework (Beck *et al.* 2018), which extends ASP with window operators. However, our focus differs: while *lars* reasons over data streams, our work targets general metric temporal reasoning. This distinction is also reflected in the semantic foundations: we rely on Metric Equilibrium Logic, while *lars* utilizes a reduct-based semantics. Similarly, metric extensions in Datalog (Wałega *et al.* 2019) have led to systems such as *meteor* (Wang *et al.* 2022) and (temporal) *vadalog* (Bellomarini *et al.* 2022). These extensions are monotonic and thus incapable of expressing default reasoning. Moreover, they are optimized for large-scale data querying with continuous time, making them less suitable for the search-intensive problems encountered in ASP. Interestingly, Wałega *et al*. (2021) investigate metric Datalog with negation under the stable model semantics, employing an HT-based semantics to define a restricted class of metric logic programs. In the spirit of Datalog, these programs limit the use of logical and temporal operators, disallowing disjunction and "existential" operators (such as eventually, since, or until) in rule heads.

Downloaded from https://www.cambridge.org/core. UNIVERSIDAD AUTONOMA METROPOLITANA, on 08 Nov 2026 at 14:47:57, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://doi.org/10.1017/S1471068426100714

Finally, Hustadt *et al.* (2020) defines reductions of (monotonic) Metric temporal logic to Linear temporal logic. A comprehensive comparative account of metric approaches in logic programming is given in Becker *et al.* (2024).

In the context of planning, Son *et al.* (2004) explore actions with durations in plain ASP. While the proposed action language deals with quantitative temporal constraints, it does not offer any metric logic foundations. Finally, outside of ASP, Cresswell and Coddington (2003) introduce durative actions and deadlines in PDDL 2.1, though deadlines were modeled indirectly via special encodings rather than as native constructs. Gerevini *et al.* (2009) later extend this language to express deadlines for any state or condition via hard or soft constraints in PDDL 3. While these approaches are tailored to specific PDDL constructs, our objective is to provide a general logical framework for metric reasoning in ASP.

## 2 Background

To ease the formal elaboration of our translations from metric ASP to regular logic programs and their extension with difference constraints, we put ourselves into the context of the logical framework of Here-and-There and Equilibrium Logic.

### *2.1 The logic of here-and-there (Pearce 1997)*

A formula over an alphabet $\mathcal{A}$ is defined as

$$\varphi ::= \bot \mid a \mid \varphi \wedge \varphi \mid \varphi \vee \varphi \mid \varphi \rightarrow \varphi \quad \text{where } a \in \mathcal{A} \ .$$

We define the derived operators $\neg\varphi = \varphi \rightarrow \bot$ and $\top = \neg\bot$. Elements $a$ of $\mathcal{A}$ are called (Boolean) *atoms*. A *literal* is an atom or an atom preceded by negation, viz. $a$ or $\neg a$. A *theory* is a set of formulas. We sometimes write $\varphi \leftarrow \psi$ instead of $\psi \rightarrow \varphi$ to follow logic programming conventions. A *program* is a set of implications of the form $\varphi \leftarrow \psi$ where $\varphi$ is a disjunction of literals and $\psi$ is a conjunction of literals.

We represent an *interpretation* $T$ as a set of atoms $T \subseteq \mathcal{A}$. An HT-*interpretation* is a pair $\langle H, T\rangle$ of interpretations such that $H \subseteq T$; it is said to be *total* if $H = T$. An HT-interpretation $\langle H, T\rangle$ *satisfies* a formula $\varphi$, written $\langle H, T\rangle \models \varphi$, if

1. $\langle H, T\rangle \models a$ if $a \in H$
2. $\langle H, T\rangle \models \varphi \wedge \psi$ if $\langle H, T\rangle \models \varphi$ and $\langle H, T\rangle \models \psi$
3. $\langle H, T\rangle \models \varphi \vee \psi$ if $\langle H, T\rangle \models \varphi$ or $\langle H, T\rangle \models \psi$
4. $\langle H, T\rangle \models \varphi \rightarrow \psi$ if $\langle H', T\rangle \not\models \varphi$ or $\langle H', T\rangle \models \psi$ for each $H' \in \{H, T\}$

An HT-interpretation $\langle H, T\rangle$ is an HT-*model* of a theory $\Gamma$ if $\langle H, T\rangle \models \varphi$ for each $\varphi \in \Gamma$. A total model $\langle T, T\rangle$ of a theory $\Gamma$ is an *equilibrium model* if there is no other model $\langle H, T\rangle$ of $\Gamma$ with $H \subset T$; $T$ is also called a *stable model* of $\Gamma$.

### *2.2 The logic of here-and-there with constraints (Cabalar et al. 2016)*

The syntax of $\mathrm{HT}_c$ relies on a signature $\langle \mathcal{X}, \mathcal{D}, \mathcal{C}\rangle$, similar to constraint satisfaction problems, where elements of set $\mathcal{X}$ represent variables and elements of $\mathcal{D}$ are domain values

Downloaded from https://www.cambridge.org/core. UNIVERSIDAD AUTONOMA METROPOLITANA, on 08 Nov 2026 at 14:47:57, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://doi.org/10.1017/S1471068426100714

(usually identified with their respective constants). The constraint atoms in $\mathcal{C}$ provide an abstract way to relate values of variables and constants according to the atom's semantics. For instance, *difference constraint atoms* are expressions of the form "$x - y \leq d$," containing variables $x, y \in \mathcal{X}$ and the domain value $d \in \mathcal{D}$. A constraint formula $\varphi$ over $\mathcal{C}$ is defined as

$$\varphi ::= \bot \mid c \mid \varphi \wedge \varphi \mid \varphi \vee \varphi \mid \varphi \rightarrow \varphi \quad \text{where } c \in \mathcal{C}$$

Concepts like defined operators, programs, theories, etc. are analogous to HT. Variables can be assigned some value from $\mathcal{D}$ or left *undefined*. For the latter, we use the special symbol $\boldsymbol{u} \notin \mathcal{D}$ and the extended domain $\mathcal{D}_u = \mathcal{D} \cup \{\boldsymbol{u}\}$. A *valuation* $v$ is a function $v : \mathcal{X} \rightarrow \mathcal{D}_u$. We let $\mathbb{V}$ stand for the set of all valuations. We sometimes represent a valuation $v$ as a set $\{(x, v(x)) \mid x \in \mathcal{X}, v(x) \in \mathcal{D}\}$ of pairs, so that $(x, \boldsymbol{u})$ is never formed. This allows us to use standard set inclusion, $v \subseteq v'$, for comparing $v, v' \in \mathbb{V}$.

The semantics of constraint atoms is defined in $\mathrm{HT}_c$ via *denotations*, that is, functions $[\![ \cdot ]\!] : \mathcal{C} \rightarrow 2^{\mathbb{V}}$ mapping each constraint atom to a set of valuations. An $\mathrm{HT}_c$-*interpretation* is a pair $\langle h, t \rangle$ of valuations such that $h \subseteq t$; it is *total* if $h = t$. Given a denotation $[\![ \cdot ]\!]$, an $\mathrm{HT}_c$-interpretation $\langle h, t \rangle$ *satisfies* a constraint formula $\varphi$, written $\langle h, t \rangle \models \varphi$, if[2]

1. $\langle h, t \rangle \models c$ if $h \in [\![ c ]\!]$
2. $\langle h, t \rangle \models \varphi \wedge \psi$ if $\langle h, t \rangle \models \varphi$ and $\langle h, t \rangle \models \psi$
3. $\langle h, t \rangle \models \varphi \vee \psi$ if $\langle h, t \rangle \models \varphi$ or $\langle h, t \rangle \models \psi$
4. $\langle h, t \rangle \models \varphi \rightarrow \psi$ if $\langle v, t \rangle \not\models \varphi$ or $\langle v, t \rangle \models \psi$ for each $v \in \{h, t\}$

An $\mathrm{HT}_c$-interpretation $\langle h, t \rangle$ is an $\mathrm{HT}_c$-*model* of a theory $\Gamma$ if $\langle h, t \rangle \models \varphi$ for every $\varphi \in \Gamma$. A total model $\langle t, t \rangle$ of a theory $\Gamma$ is a *constraint equilibrium model* if there is no other model $\langle h, t \rangle$ of $\Gamma$ with $h \subset t$.

### *2.3 The metric temporal logic of here-and-there (Cabalar et al. 2020b; Becker et al. 2024)*

Given an alphabet $\mathcal{A}$ and $\mathbb{I} = \{[m..n) \mid m \in \mathbb{N}, n \in \mathbb{N} \cup \{\omega\}\}$, a metric temporal formula is defined as[3]

$$\varphi ::= \bot \mid a \mid \varphi \wedge \varphi \mid \varphi \vee \varphi \mid \varphi \rightarrow \varphi \mid \mathbf{I} \mid \bigcirc_I \varphi \mid \Box_I \varphi \mid \Diamond_I \varphi \quad \text{where } a \in \mathcal{A}, I \in \mathbb{I}$$

The last three cases deal with metric temporal operators, which are indexed by some interval $I$. In words, $\bigcirc_I$, $\Box_I$, and $\Diamond_I$ are called *next*, *always*, and *eventually*. $\mathbf{I}$ simply refers to the initial state. We write $\bigcirc$, $\Box$, $\Diamond$ for $\bigcirc_{[0..\omega)}$, $\Box_{[0..\omega)}$, $\Diamond_{[0..\omega)}$, respectively. In addition to the aforedefined Boolean operators, we define $\mathbf{F} = \neg\bigcirc\top$, which allow us to refer to the final state. Concepts like programs, theories, etc. are analogous to HT.

The semantics of temporal formulas is defined via *traces*, being sequences $\mathbf{T} = (T_i)_{i \in [0..\lambda)}$ of interpretations $T_i \subseteq \mathcal{A}$; $\lambda$ is the length of $\mathbf{T}$. Here, we consider only traces of finite length. We define the ordering $\mathbf{H} \leq \mathbf{T}$ between traces of the same length $\lambda$ as

[2] Since we use strict denotations, it is sufficient to check in Condition 1 satisfaction in $h$.
[3] More general formulas, including until, release and past operators, are presented in Becker *et al.* (2024).

Downloaded from https://www.cambridge.org/core. UNIVERSIDAD AUTONOMA METROPOLITANA, on 08 Nov 2026 at 14:47:57, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://doi.org/10.1017/S1471068426100714

$H_i \subseteq T_i$ for each $i \in [0..\lambda)$, and $\mathbf{H} < \mathbf{T}$ as both $\mathbf{H} \leq \mathbf{T}$ and $\mathbf{H} \neq \mathbf{T}$. An HT-*trace* over $\mathcal{A}$ of length $\lambda$ is a sequence of pairs $(\langle H_i, T_i\rangle)_{i\in[0..\lambda)}$ with $H_i \subseteq T_i$ for any $0 \leq i < \lambda$; it is *total* if $H_i = T_i$. For convenience, we represent it as the pair $\langle \mathbf{H}, \mathbf{T}\rangle$ of traces $\mathbf{H} = (H_i)_{i\in[0..\lambda)}$ and $\mathbf{T} = (T_i)_{i\in[0..\lambda)}$.

Metric information is captured by timing functions. Given $\lambda \in \mathbb{N}$, we say that $\tau : [0..\lambda) \to \mathbb{N}$ is a (strict) *timing function* wrt $\lambda$ if $\tau(0) = 0$ and $\tau(k) < \tau(k+1)$ for $0 \leq k < \lambda - 1$. A *timed* HT-trace $(\langle \mathbf{H}, \mathbf{T}\rangle, \tau)$ over $\mathcal{A}$ and $(\mathbb{N}, <)$ of length $\lambda$ is a pair consisting of an HT-trace $\langle \mathbf{H}, \mathbf{T}\rangle$ over $\mathcal{A}$ of length $\lambda$ and a timing function $\tau$ wrt $\lambda$. A timed HT-trace $\mathbf{M} = (\langle \mathbf{H}, \mathbf{T}\rangle, \tau)$ of length $\lambda$ over alphabet $\mathcal{A}$ *satisfies* a metric formula $\varphi$ at $k \in [0..\lambda)$, written $\mathbf{M}, k \models \varphi$, if

1. $\mathbf{M}, k \models a$ if $a \in H_k$
2. $\mathbf{M}, k \models \varphi \wedge \psi$ if $\mathbf{M}, k \models \varphi$ and $\mathbf{M}, k \models \psi$
3. $\mathbf{M}, k \models \varphi \vee \psi$ if $\mathbf{M}, k \models \varphi$ or $\mathbf{M}, k \models \psi$
4. $\mathbf{M}, k \models \varphi \rightarrow \psi$ if $\mathbf{M}', k \not\models \varphi$ or $\mathbf{M}', k \models \psi$, for both $\mathbf{M}' = \mathbf{M}$ and $\mathbf{M}' = (\langle \mathbf{T}, \mathbf{T}\rangle, \tau)$
5. $\mathbf{M}, k \models \mathbf{I}$ if $k = 0$
6. $\mathbf{M}, k \models \bigcirc_I \varphi$ if $k + 1 < \lambda$ and $\mathbf{M}, k{+}1 \models \varphi$ and $\tau(k{+}1) - \tau(k) \in I$
7. $\mathbf{M}, k \models \Diamond_I \varphi$ if $\mathbf{M}, i \models \varphi$ for some $i$ with $k \leq i < \lambda$ and $\tau(i) - \tau(k) \in I$
8. $\mathbf{M}, k \models \Box_I \varphi$ if $\mathbf{M}, i \models \varphi$ for all $i$ with $k \leq i < \lambda$ and $\tau(i) - \tau(k) \in I$

A timed HT-trace $\mathbf{M}$ is an *MHT-model* of a metric theory $\Gamma$ if $\mathbf{M}, 0 \models \varphi$ for all $\varphi \in \Gamma$. A total MHT-model $(\langle \mathbf{T}, \mathbf{T}\rangle, \tau)$ of a theory $\Gamma$ is a *metric equilibrium model* if there is no other model $(\langle \mathbf{H}, \mathbf{T}\rangle, \tau)$ of $\Gamma$ with $\mathbf{H} < \mathbf{T}$.

For illustration, let us consider the formalization of the dentist scenario in equations (1) to (9). We assume that variables $L$ and $L'$ are substituted by distinct locations *office*, *atm*, *dentist*, and *home*; and variable $I$ by items *cash* and *icard*. We use $\delta\langle L, L'\rangle$ to refer to the distance between two locations from Table 1. As in Mellarkod (2007), we assume that Ram is automatically picking up items when being at the same position.

$$\Box(at(ram, office) \leftarrow \mathbf{I}) \tag{1}$$
$$\Box(at(cash, atm) \leftarrow \mathbf{I}) \tag{2}$$
$$\Box(at(icard, home) \leftarrow \mathbf{I}) \tag{3}$$
$$\Box(\bigvee_{L'\neq L} go(ram, L') \leftarrow at(ram, L) \wedge \neg\mathbf{F}) \tag{4}$$
$$\Box(has(ram, I) \leftarrow at(ram, L) \wedge at(I, L)) \tag{5}$$
$$\Box(at(I, L) \leftarrow at(ram, L) \wedge has(ram, I)) \tag{6}$$
$$\Box(\bigcirc_{[\delta\langle L,L'\rangle..\delta\langle L,L'\rangle+1)} at(ram, L') \leftarrow at(ram, L) \wedge go(ram, L')) \tag{7}$$
$$\Box(\bigcirc has(ram, I) \leftarrow has(ram, I) \wedge \neg\mathbf{F}) \tag{8}$$
$$\Box(\bigcirc at(I, L) \leftarrow \neg has(ram, I) \wedge at(I, L) \wedge \neg\mathbf{F}) \tag{9}$$

In brief, Rules equations (1) to (3) give the initial situation. Rule equation (4) delineates possible actions. Rules equations (5) and (6) capture indirect effects. Rule equation (7) is the effect of moving from location $L$ to $L'$: it uses the next operator restricted by the duration between locations. Rules equations (8) and (9) address inertia.

Downloaded from https://www.cambridge.org/core. UNIVERSIDAD AUTONOMA METROPOLITANA, on 08 Nov 2026 at 14:47:57, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://doi.org/10.1017/S1471068426100714

## 3 Plain metric logic programs

Metric logic programs, defined as metric theories composed of implications akin to logic programming rules, derive their semantics from their metric equilibrium models. Syntactically, a *plain metric logic program* over $\mathcal{A}$ is a set of *plain metric rules* of form

$$\Box(\alpha \leftarrow \beta) \quad \text{or} \quad \Box(\bigcirc_I a \leftarrow \beta)$$

for $\alpha = a_1 \vee \ldots \vee a_m \vee \neg a_{m+1} \vee \ldots \vee \neg a_n$, $\beta = b_1 \wedge \ldots \wedge b_o \wedge \neg b_{o+1} \wedge \ldots \wedge \neg b_p$, and $0 \leq m \leq n \leq o \leq p$ with $a, a_i \in \mathcal{A}$ for $1 \leq i \leq n$, and $b_i \in \mathcal{A} \cup \{\mathbf{I}, \mathbf{F}\}$ for $1 \leq i \leq p$.

While our considered language fragment excludes global temporal operators and disjunctive metric heads, it effectively captures state transitions and allows for imposing timing constraints upon them. A comprehensive treatment is provided in Section 4.

Our two alternative translations share a common structure, each divided into three distinct parts. The first part maps a plain metric program into a regular one. This part captures the state transitions along an HT-trace specified by the metric program, and is common to both translations. The second and third parts capture the timing function along with its interplay with the interval constraints imposed by the metric program, respectively. The two variants of these parts are described in Sections 3.1 and 3.2 below.

The first part of our translation takes a plain metric program over $\mathcal{A}$ and yields rules over $\mathcal{A}^* = \bigcup_{k \in \mathbb{N}} \mathcal{A}_k$ for $\mathcal{A}_k = \{a_k \mid a \in \mathcal{A}\}$ and $k \in \mathbb{N}$. Atoms of form $a_k$ in $\mathcal{A}^*$ represent the values taken by variable $a \in \mathcal{A}$ at different points $k$ along a trace of length $\lambda$.

We begin by inductively defining the translation $(r)_k$ of a plain metric rule $\Box r$ at $k$ for $0 \leq k < \lambda$ and $\lambda \in \mathbb{N}$ as follows:[4]

$$\begin{aligned}
(\bot)_k &= \bot \\
(a)_k &= a_k \text{ if } a \in \mathcal{A} \\
(\bigcirc_I a)_k &= \begin{cases} \bot & \text{if } k = \lambda - 1 \\ (a)_{k+1} & \text{otherwise} \end{cases} \\
(\mathbf{I})_k &= \begin{cases} \top & \text{if } k = 0 \\ \bot & \text{otherwise} \end{cases} \\
(\varphi_1 \wedge \varphi_2)_k &= (\varphi_1)_k \wedge (\varphi_2)_k \\
(\varphi_1 \vee \varphi_2)_k &= (\varphi_1)_k \vee (\varphi_2)_k \\
(\alpha \leftarrow \beta)_k &= \{(\varphi_1)_k \leftarrow (\varphi_2)_k\}
\end{aligned}$$

Note that we drop the always operator $\Box$ preceding rules in the translation; it is captured by producing a rule instance for every $0 \leq k < \lambda$. Accordingly, for a plain metric program $P$ over $\mathcal{A}$ and $\lambda \in \mathbb{N}$, we define

$$\Pi_\lambda(P) = \bigcup_{\Box r \in P,\, 0 \leq k < \lambda} (r)_k \quad \text{over } \mathcal{A}^*.$$

For illustration, consider the instance of equation (7) for moving from *office* to *home*

$$\Box(\bigcirc_{[15..16)} at(ram, home) \leftarrow at(ram, office) \wedge go(ram, home)) \ . \tag{10}$$

[4] Note that $\top$, $\neg$ and $\mathbf{F}$ are defined operators.

Downloaded from https://www.cambridge.org/core. UNIVERSIDAD AUTONOMA METROPOLITANA, on 08 Nov 2026 at 14:47:57, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://doi.org/10.1017/S1471068426100714

Our translation ignores $\Box$ at first and yields:

$$\bot \leftarrow at(ram, office)_k \wedge go(ram, home)_k \quad \text{for } k = \lambda - 1 \tag{11}$$

$$at(ram, home)_{k+1} \leftarrow at(ram, office)_k \wedge go(ram, home)_k \quad \text{otherwise} \tag{12}$$

When assembling $\Pi_\lambda(P)$ for $\lambda = 100$ and $P$ being the rules in equations (1) to (9), we account for $\Box$ by adding 99 instances of the rule in equation (12) and a single instance of equation (11).

This first part of our translation follows Kamp's translation (Kamp 1968) for Linear temporal logic. Of particular interest is the translation of $\bigcirc_I a \leftarrow \beta$. The case analysis accounts for the actual state transition of the next operator, which is infeasible at the end of the trace. Thus, we either derive $a_{k+1}$ or a contradiction. The metric aspect is captured by the translations in Section 3.1 and 3.2. Whenever all intervals in a plain metric programs $P$ are of form $[0..\omega)$, we get a one-to-one correspondence between MHT traces of length $\lambda$ of $P$ with an arbitrary yet fixed timing function and HT-interpretations of $\Pi_\lambda(P)$. Finally, it is worth noting that the size of the resulting program $\Pi_\lambda(P)$ grows with $\lambda$.

### *3.1 Translation of plain metric logic programs to HT*

We begin by formalizing timing functions $\tau$ via Boolean atoms in $\mathcal{T} = \{t_{k,d} \mid k, d \in \mathbb{N}\}$. An atom like $t_{k,d}$ represents that $\tau(k) = d$. To obtain finite theories, we furthermore impose an upper bound $\nu \in \mathbb{N}$ on the range of $\tau$. Hence, together with the trace length $\lambda$, our formalization $\Delta_{\lambda,\nu}$ only captures timing functions $\tau$ satisfying $\tau(\lambda - 1) \leq \nu$.

Given $\lambda, \nu \in \mathbb{N}$, we let

$$\Delta_{\lambda,\nu} = \{t_{0,0}\} \cup \{\textstyle\bigvee_{d<d' \leq \nu} t_{k+1,d'} \leftarrow t_{k,d} \mid 0 \leq k < \lambda - 1, 0 \leq d \leq \nu\} \tag{13}$$

Starting from $\tau(0) = 0$, represented by $t_{0,0}$, the rule in equation (13) assigns strictly increasing time points to consecutive states, reflecting that $\tau(k) < \tau(k+1)$ for $0 \leq k < \lambda - 1$.

The last part of our formalization accounts for the interplay of the timing function with the interval conditions imposed in the program. Given $\lambda, \nu \in \mathbb{N}$ and a plain metric program $P$, we let

$$\Psi_{\lambda,\nu}(P) = \{\bot \leftarrow (\beta)_k \wedge t_{k,d} \wedge t_{k+1,d'} \mid 0 \leq k < \lambda - 1, 0 \leq d < d' \leq \nu, \tag{14}$$
$$d' - d < m, \Box(\textstyle\bigcirc_{[m..n)} a \leftarrow \beta) \in P\} \cup$$

$$\{\bot \leftarrow (\beta)_k \wedge t_{k,d} \wedge t_{k+1,d'} \mid 0 \leq k < \lambda - 1, 0 \leq d < d' \leq \nu, \tag{15}$$
$$n \in \mathbb{N}, d' - d \geq n, \Box(\textstyle\bigcirc_{[m..n)} a \leftarrow \beta) \in P\}$$

The integrity constraints ensure that for every plain metric rule $\Box(\bigcirc_{[m..n)} a \leftarrow \beta)$ the duration between the $k$th and $(k+1)$st state in a trace falls within interval $[m..n)$. With $\tau(k) = d$ and $\tau(k+1) = d'$, this amounts to checking whether $d + m \leq d' < d + n$, if $n$ is finite; otherwise, the verification of the upper bound in equation (15) is dropped for $n = \omega$.

Downloaded from https://www.cambridge.org/core. UNIVERSIDAD AUTONOMA METROPOLITANA, on 08 Nov 2026 at 14:47:57, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://doi.org/10.1017/S1471068426100714

Note that the size of both $\Delta_{\lambda,\nu}$ and $\Psi_{\lambda,\nu}(P)$ is proportional to $\mathcal{O}(\lambda \cdot \nu^2)$. Hence, long traces and even more severely fine-grained timing functions lead to a significant blow up when translating plain metric programs into regular ones with the above formalization.

For the rule in equation (10), we get

$$\bot \leftarrow at(ram, office)_k \wedge go(ram, home)_k \wedge t_{k,d} \wedge t_{k+1,d'} \quad \text{for } d' - d < 15 \tag{16}$$

$$\bot \leftarrow at(ram, office)_k \wedge go(ram, home)_k \wedge t_{k,d} \wedge t_{k+1,d'} \quad \text{for } d' - d \geq 16 \tag{17}$$

and $0 \leq k < \lambda - 1$, $0 \leq d < d' \leq \nu$. For $\lambda = 100$ and $\nu = 1000$, this then amounts to roughly $10^8$ instances for each of the above constraints.

In what follows, we characterize the effect of our formalization in terms of HT-models, and ultimately show the completeness and correctness of our translation.

*Definition 1.*
An HT interpretation $\langle H, T\rangle$ over $\mathcal{A}$ with $\mathcal{T} \subseteq \mathcal{A}$, is timed wrt $\lambda \in \mathbb{N}$,

if there is a timing function $\tau$ wrt $\lambda$ such that for all $0 \leq k < \lambda$, $d \in \mathbb{N}$, we have

$$\langle H, T\rangle \models t_{k,d} \text{ iff } \tau(k) = d \quad \text{and} \quad \langle T, T\rangle \models t_{k,d} \text{ iff } \tau(k) = d$$

We also call $\tau$ the timing function induced by $\langle H, T\rangle$.

*Proposition 1.*
Let $\lambda, \nu \in \mathbb{N}$.

If $\langle T, T\rangle$ is an equilibrium model of $\Delta_{\lambda,\nu}$ then $\langle T, T\rangle$ is timed wrt $\lambda$.

*Proposition 2.*
Let $\langle H, T\rangle$ be an HT interpretation and $\lambda \in \mathbb{N}$.

If $\langle H, T\rangle$ is timed wrt $\lambda$ and induces $\tau$, then $\langle H, T\rangle$ is an HT-model of $\Delta_{\lambda,\nu}$ for $\nu = \tau(\lambda - 1)$.

Clearly, the last proposition extends to equilibrium models.

Given a timed HT-trace $\mathbf{M} = (\langle H_k, T_k\rangle_{k\in[0..\lambda)}, \tau)$ of length $\lambda$ over $\mathcal{A}$, we define $\theta(\mathbf{M})$ as HT interpretation $\langle H \cup X, T \cup X\rangle$ where

$$H = \{a_k \in \mathcal{A}_k \mid 0 \leq k < \lambda, a \in H_k\} \qquad T = \{a_k \in \mathcal{A}_k \mid 0 \leq k < \lambda, a \in T_k\}$$
$$X = \{t_{k,d} \mid \tau(k) = d, 0 \leq k < \lambda, d \in \mathbb{N}\}$$

Note that $\theta(\mathbf{M})$ is an HT interpretation timed wrt $\lambda$.

Conversely, given an HT interpretation $\langle H, T\rangle$ timed wrt $\lambda$ over $\mathcal{A}^* \cup \mathcal{T}$ and its induced timing function $\tau$, we define $\sigma(\langle H, T\rangle)$ as the timed HT-trace

$$(\langle \{a \in \mathcal{A} \mid a_k \in H\}, \{a \in \mathcal{A} \mid a_k \in T\}\rangle_{k\in[0..\lambda)}, \tau)$$

In fact, both functions $\sigma$ and $\theta$ are invertibles, and we get a one-to-one correspondence between HT interpretations timed wrt $\lambda$ and timed HT-traces of length $\lambda$.

Finally, we have the following completeness and correctness result.

*Theorem 1 (Completeness).*
Let $P$ be a plain metric logic program and $\mathbf{M} = (\langle \mathbf{T}, \mathbf{T}\rangle, \tau)$ a total timed HT-trace of length $\lambda$.

Downloaded from https://www.cambridge.org/core. UNIVERSIDAD AUTONOMA METROPOLITANA, on 08 Nov 2026 at 14:47:57, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://doi.org/10.1017/S1471068426100714

If $\mathbf{M}$ is a metric equilibrium model of $P$, then $\theta(\mathbf{M})$ is an equilibrium model of $\Pi_\lambda(P) \cup \Delta_{\lambda,\nu} \cup \Psi_{\lambda,\nu}(P)$ with $\nu = \tau(\lambda - 1)$.

*Theorem 2 (Correctness).*
Let $P$ be a plain metric logic program, and $\lambda, \nu \in \mathbb{N}$.

If $\langle T, T\rangle$ is an equilibrium model of $\Pi_\lambda(P) \cup \Delta_{\lambda,\nu} \cup \Psi_{\lambda,\nu}(P)$, then $\sigma(\langle T, T\rangle)$ is a metric equilibrium model of $P$.

### *3.2 Translation of plain metric logic programs to HT$_c$*

We now present an alternative, refined formalization of the second and third parts, utilizing integer variables and difference constraints to capture the timing function more effectively. To this end, we use the logic of HT$_c$ to combine the Boolean nature of ASP with constraints on integer variables.

Given base alphabet $\mathcal{A}$ and $\lambda \in \mathbb{N}$, we consider the HT$_c$ signature $\langle \mathcal{X}, \mathcal{D}, \mathcal{C}\rangle$ where[5]

$$\begin{aligned}
\mathcal{X} &= \mathcal{A}^* \cup \{t_k \mid 0 \leq k < \lambda\} \\
\mathcal{D} &= \{\boldsymbol{t}\} \cup \mathbb{N} \\
\mathcal{C} &= \{a = \boldsymbol{t} \mid a \in \mathcal{A}^*\} \cup \{x = d \mid x \in \mathcal{X} \setminus \mathcal{A}^*, d \in \mathbb{N}\} \cup \\
&\quad \{x - y \leq d \mid x, y \in \mathcal{X} \setminus \mathcal{A}^* \in \mathbb{N}, d \in \mathbb{Z}\}
\end{aligned}$$

Rather than using Boolean variables, this signature represents timing functions $\tau$ directly by integer variables $t_k$, capturing that $\tau(k) = t_k$ for $0 \leq k < \lambda$. This is enforced by the integer constraints in $\mathcal{C}$, whose meaning is defined by the following denotations:

$$\begin{aligned}
[\![\, a = \boldsymbol{t} \,]\!] &= \{v \in \mathbb{V} \mid v(a) = \boldsymbol{t}\} && \text{for all } a \in \mathcal{A}^* \\
[\![\, x = d \,]\!] &= \{v \in \mathbb{V} \mid v(x), d \in \mathbb{N},\ v(x) = d\} \\
[\![\, x - y \leq d \,]\!] &= \{v \in \mathbb{V} \mid v(x), v(y) \in \mathbb{N}, d \in \mathbb{Z},\ v(x) - v(y) \leq d\}
\end{aligned}$$

When dealing with Boolean variables, we simplify notation by representing $a = \boldsymbol{t}$ as $a$.

This leads us to the following counterpart of $\Delta_{\lambda,\nu}$ in equation (13). Given $\lambda \in \mathbb{N}$, we define

$$\Delta^c_\lambda = \{t_0 = 0\} \cup \{t_k - t_{k+1} \leq -1 \mid 0 \leq k < \lambda - 1\} \tag{18}$$

Starting from $t_0 = 0$, the difference constraints in equation (18) enforce that $t_k < t_{k+1}$ reflecting that $\tau(0) = 0$ and $\tau(k) < \tau(k+1)$ for $0 \leq k < \lambda - 1$. Moreover, $\Delta^c_\lambda$ is unbound and thus imposes no restriction on timing functions. And no variable $t_k$ is ever undefined:

*Proposition 3.*
Let $\langle h, t\rangle$ be an HT$_c$ interpretation and $\lambda \in \mathbb{N}$

If $\langle h, t\rangle$ is an HT$_c$-model of $\Delta^c_\lambda$, then $h(t_k) \in \mathbb{N}$ for all $0 \leq k < \lambda$.

We also have $t(t_k) \in \mathbb{N}$ by definition of HT$_c$ interpretations, that is, since $h \subseteq t$.

Our variant of the third part of our translation re-expresses the ones in equations (14/15) in terms of integer variables and difference constraints. Given $\lambda \in \mathbb{N}$ and a plain

[5] In HT$_c$ (Cabalar *et al.* 2016), Boolean variables are already captured by truth values $\boldsymbol{t}$ and $\boldsymbol{u}$ (rather than $\boldsymbol{f}$[alse]).

Downloaded from https://www.cambridge.org/core. UNIVERSIDAD AUTONOMA METROPOLITANA, on 08 Nov 2026 at 14:47:57, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://doi.org/10.1017/S1471068426100714

metric logic program $P$, we define

$$\Psi^c_\lambda(P) = \{\bot \leftarrow (\beta)_k \wedge \neg(t_k - t_{k+1} \leq -m) \mid 0 \leq k < \lambda - 1, \quad \Box(\bigcirc_{[m..n)} a \leftarrow \beta) \in P\} \cup \tag{19}$$

$$\{\bot \leftarrow (\beta)_k \wedge \neg(t_{k+1} - t_k \leq n - 1) \mid 0 \leq k < \lambda - 1, \quad n \in \mathbb{N}, \Box(\bigcirc_{[m..n)} a \leftarrow \beta) \in P\} \tag{20}$$

In fact, both $\Delta^c_\lambda$ and $\Psi^c_\lambda(P)$ drop the upper bound on the range of a timing function, as required in their Boolean counterparts. Hence, their size is only proportional to $\mathcal{O}(\lambda)$, and thus considerably smaller than their purely Boolean counterparts.

For the rule in equation (10), we get

$$\bot \leftarrow \mathit{at}(\mathit{ram}, \mathit{office})_k \wedge \mathit{go}(\mathit{ram}, \mathit{home})_k \wedge \neg(t_k - t_{k+1} \leq -15) \tag{21}$$

$$\bot \leftarrow \mathit{at}(\mathit{ram}, \mathit{office})_k \wedge \mathit{go}(\mathit{ram}, \mathit{home})_k \wedge \neg(t_{k+1} - t_k \leq 15) \tag{22}$$

for $0 \leq k < \lambda - 1$. Given $\lambda = 100$, this only amounts to $10^2$ instances.

Mirroring our approach in Section 3.1, we capture the meaning of $\Delta^c_\lambda$ using specialized $\mathrm{HT}_c$-models. This leads to the completeness and correctness of our translation.

*Definition 2.*
An $\mathrm{HT}_c$ interpretation $\langle h, t\rangle$ over $\langle \mathcal{X}, \mathcal{D}, \mathcal{C}\rangle$ is timed wrt $\lambda$, if there is a timing function $\tau$ wrt $\lambda$ such that $h(t_k) = \tau(k)$ and $t(t_k) = \tau(k)$ for all $0 \leq k < \lambda$.

As above, we call $\tau$ the timing function induced by $\langle h, t\rangle$.

*Proposition 4.*
Let $\langle h, t\rangle$ be an $\mathrm{HT}_c$ interpretation and $\lambda \in \mathbb{N}$.

If $\langle h, t\rangle$ is an $\mathrm{HT}_c$-model of $\Delta^c_\lambda$ then $\langle h, t\rangle$ is timed wrt $\lambda$.

*Proposition 5.*
Let $\langle h, t\rangle$ be an $\mathrm{HT}_c$ interpretation and $\lambda \in \mathbb{N}$.

If $\langle h, t\rangle$ is timed wrt $\lambda$ then $\langle h, t\rangle$ is an $\mathrm{HT}_c$-model of $\Delta^c_\lambda$.

Unlike Proposition 1, the latter refrain from requiring $\mathrm{HT}_c$-interpretations in equilibrium.

Given an HT-trace $\mathbf{M} = (\langle H_k, T_k\rangle_{0 \leq k < \lambda}, \tau)$ of length $\lambda$, we define $\theta^c(\mathbf{M})$ as the $\mathrm{HT}_c$ interpretation $\langle h \cup x, t \cup x\rangle$ where $h, t, x$ are valuations such that

$$h = \{(a_k, \boldsymbol{t}) \mid 0 \leq k < \lambda, a \in H_k\} \qquad t = \{(a_k, \boldsymbol{t}) \mid 0 \leq k < \lambda, a \in T_k\}$$
$$x = \{(t_k, d) \mid \tau(k) = d, 0 \leq k < \lambda, d \in \mathbb{N}\}$$

Similar to above, $\theta^c(\mathbf{M})$ is an $\mathrm{HT}_c$ interpretation timed wrt $\lambda$.

Conversely, given an $\mathrm{HT}_c$ interpretation $\langle h, t\rangle$ timed wrt $\lambda$ and its induced timing function $\tau$, we define $\sigma^c(\langle h, t\rangle)$ as the timed HT-trace

$$(\langle \{a \mid h(a_k) = \boldsymbol{t}\}, \{a \mid t(a_k) = \boldsymbol{t}\}\rangle_{0 \leq k < \lambda}, \tau)$$

As above, functions $\sigma^c$ and $\theta^c$ are invertibles. Hence, we get a one-to-one correspondence between $\mathrm{HT}_c$ interpretations timed wrt $\lambda$ and timed HT-traces of length $\lambda$.

Finally, we have the following completeness and correctness result.

Downloaded from https://www.cambridge.org/core. UNIVERSIDAD AUTONOMA METROPOLITANA, on 08 Nov 2026 at 14:47:57, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://doi.org/10.1017/S1471068426100714

*Theorem 3 (Completeness).*
Let $P$ be a plain metric logic program and $\mathbf{M} = (\langle \mathbf{T}, \mathbf{T} \rangle, \tau)$ a total timed HT-trace of length $\lambda$.

If $\mathbf{M}$ is a metric equilibrium model of $P$, then $\theta^c(\mathbf{M})$ is a constraint equilibrium model of $\Pi_\lambda(P) \cup \Delta^c_\lambda \cup \Psi^c_\lambda(P)$.

*Theorem 4 (Correctness).*
Let $P$ be a plain metric logic program, and $\lambda \in \mathbb{N}$.

If $\langle t, t \rangle$ is a constraint equilibrium model of $\Pi_\lambda(P) \cup \Delta^c_\lambda \cup \Psi^c_\lambda(P)$, then $\sigma^c(\langle t, t \rangle)$ is a metric equilibrium model of $P$.

## 4 General metric logic programs

In the preceding section, we introduced a simplified fragment of metric theories and demonstrated its application in modeling quantitatively constrained transitions for dynamic problems. While this fragment enables the representation of basic planning and scheduling scenarios, it lacks the capacity to express variable queries involving global temporal operators.

In this section, we show how general metric theories can be reduced to a more general logic programming format and how rules in this format can be implemented in analogy to the techniques shown in the previous section. As in Section 3, we define metric logic programs as metric theories composed of implications analogous to logic programming rules, with their semantics determined by their metric equilibrium models.

Syntactically, a (general) *metric logic program* over $\mathcal{A}$ is a set of (general) *metric rules* of form

$$\Box(\alpha \leftarrow \beta) \tag{23}$$

for $\alpha = \mu_1 \vee \ldots \vee \mu_m \vee \neg\mu_{m+1} \vee \ldots \vee \neg\mu_n$, $\beta = \mu_{n+1} \wedge \ldots \wedge \mu_o \wedge \neg\mu_{o+1} \wedge \ldots \wedge \neg\mu_p$, and $0 \leq m \leq n \leq o \leq p$, where for $1 \leq i \leq p$ each $\mu_i$ is a *metric atom* over $\mathcal{A}$ defined as

$$\mu ::= \bot \mid \top \mid a \in \mathcal{A} \mid \mathbf{I} \mid \bigcirc_I b \mid \Box_I b \mid \Diamond_I b \quad \text{where } b \in \mathcal{A} \cup \{\bot, \top\} \text{ and } I \in \mathbb{I}$$

We include the truth constants, for instance, to express via $\Diamond_I \top$ that there is a state within interval $I$ or to define $\mathbf{F}$ as $\neg\bigcirc\top$. We write $\varphi \in r$ whenever a metric formula $\varphi$ occurs in a rule $r$ as in equation (23), that is, if $\varphi = \mu_i$ for some $1 \leq i \leq p$, or $\varphi = b$ where $\mu_i \in \{\bigcirc_I b, \Diamond_I b, \Box_I b\}$ for some $1 \leq i \leq p$.

Note that metric atoms only apply temporal operators to positive atoms and Boolean constants. The next result shows that this is no restriction.

*Proposition 6.*
Given a metric formula $\varphi$, we have the following equivalences

1. $\Diamond_I \neg\varphi \equiv \neg\Box_I \varphi$
2. $\Box_I \neg\varphi \equiv \neg\Diamond_I \varphi$
3. $\bigcirc_I \neg\varphi \equiv \bigcirc_I \top \wedge \neg\bigcirc_I \varphi$

To reduce arbitrary metric formulas to a metric program, we define a Tseitin-style translation (Tseitin 1968) that replaces each subformula with a corresponding fresh atom.

Downloaded from https://www.cambridge.org/core. UNIVERSIDAD AUTONOMA METROPOLITANA, on 08 Nov 2026 at 14:47:57, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://doi.org/10.1017/S1471068426100714

Table 2. *Definition of sets $\eta(\phi)$ and $\eta^*(\phi)$ for any metric temporal formula $\phi$*

| $\phi$ | $\eta(\phi)$ | $\eta^*(\phi)$ |
|---|---|---|
| $a$ | $\Box(L_a \leftrightarrow a)$ | $\Box(L_a \rightarrow a)$<br>$\Box(a \rightarrow L_a)$ |
| $\varphi \wedge \psi$ | $\Box(L_{\varphi\wedge\psi} \leftrightarrow L_\varphi \wedge L_\psi)$ | $\Box(L_{\varphi\wedge\psi} \rightarrow L_\varphi)$<br>$\Box(L_{\varphi\wedge\psi} \rightarrow L_\psi)$<br>$\Box(L_\varphi \wedge L_\psi \rightarrow L_{\varphi\wedge\psi})$ |
| $\varphi \vee \psi$ | $\Box(L_{\varphi\vee\psi} \leftrightarrow L_\varphi \vee L_\psi)$ | $\Box(L_\varphi \rightarrow L_{\varphi\vee\psi})$<br>$\Box(L_\psi \rightarrow L_{\varphi\vee\psi})$<br>$\Box(L_{\varphi\vee\psi} \rightarrow L_\varphi \vee L_\psi)$ |
| $\varphi \rightarrow \psi$ | $\Box(L_{\varphi\rightarrow\psi} \leftrightarrow L_\varphi \rightarrow L_\psi)$ | $\Box(\neg L_\varphi \rightarrow L_{\varphi\rightarrow\psi})$<br>$\Box(L_\psi \rightarrow L_{\varphi\rightarrow\psi})$<br>$\Box(L_{\varphi\rightarrow\psi} \wedge L_\varphi \rightarrow L_\psi)$<br>$\Box(\top \rightarrow L_\varphi \vee \neg L_\psi \vee L_{\varphi\rightarrow\psi})$ |
| $\bot$ | $\Box(L_\bot \leftrightarrow \bot)$ | $\Box(L_\bot \rightarrow \bot)$ |
| $\mathbf{I}$ | $\Box(L_\mathbf{I} \leftrightarrow \mathbf{I})$ | $\Box(L_\mathbf{I} \rightarrow \mathbf{I})$<br>$\Box(\mathbf{I} \rightarrow L_\mathbf{I})$ |
| $\bigcirc_I \varphi$ | $\Box(L_{\bigcirc_I\varphi} \leftrightarrow \bigcirc_I L_\varphi)$ | $\Box(L_{\bigcirc_I\varphi} \rightarrow \bigcirc_I L_\varphi)$<br>$\Box(\bigcirc_I L_\varphi \rightarrow L_{\bigcirc_I\varphi})$ |
| $\Diamond_I \varphi$ | $\Box(L_{\Diamond_I\varphi} \leftrightarrow \Diamond_I L_\varphi)$ | $\Box(L_{\Diamond_I\varphi} \rightarrow \Diamond_I L_\varphi)$<br>$\Box(\Diamond_I L_\varphi \rightarrow L_{\Diamond_I\varphi})$ |
| $\Box_I \varphi$ | $\Box(L_{\Box_I\varphi} \leftrightarrow \Box_I L_\varphi)$ | $\Box(L_{\Box_I\varphi} \rightarrow \Box_I L_\varphi)$<br>$\Box(\Box_I L_\varphi \rightarrow L_{\Box_I\varphi})$ |

To this end, we let $sub(\varphi)$ stand for the set of all subformulas of a metric formula $\varphi$ (including $\varphi$ itself). The fresh variables needed for translating $\varphi$ are given by $\mathcal{L}_\varphi = \{L_\phi \mid \phi \in sub(\varphi)\}$. With this, we define the translation of a metric formula $\varphi$ into a metric logic program $\Theta(\varphi)$ as

$$\Theta(\varphi) = \{\Box(\mathbf{I} \rightarrow L_\varphi)\} \cup \bigcup_{\phi\in sub(\varphi)} \eta^*(\phi) \tag{24}$$

The definition of the sets $\eta^*(\phi)$ is given in Table 2. Such a Tseitin-style translation adds a fresh symbol for each subformula and puts them into an equivalence. This is shown on the left hand side of Table 2. These equivalences are then translated into implications, which amount to the logic programs in the right column.

For example, consider the following metric formula enforcing the goal condition of Ram being in the office with both items within one hour.

$$(\neg\neg\Diamond_{\leq 60} goal) \wedge ((at(ram, office) \wedge has(ram, cash) \wedge has(ram, icard)) \rightarrow goal) \tag{25}$$

We apply double negation to the goal, which is equivalent to $(\Diamond_{\leq 60} goal \rightarrow \bot) \rightarrow \bot$, to ensure that this condition is inferred indirectly by other formulas, rather than directly entailed. In what follows we will abbreviate formula equation (25) as $q$ and $at(ram, office) \wedge has(ram, cash) \wedge has(ram, icard)$ as $g$ for brevity.

Downloaded from https://www.cambridge.org/core. UNIVERSIDAD AUTONOMA METROPOLITANA, on 08 Nov 2026 at 14:47:57, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://doi.org/10.1017/S1471068426100714

The metric logic program capturing the formula equation (25) is then given by:

$$\Theta(q) = \{\Box(\mathbf{I} \rightarrow L_q)\} \cup Q^*$$

where $Q^*$ is derived by translating equivalences in $Q$ into implications, as illustrated in Table 2.

$$\begin{aligned} Q = &\{\Box(L_a \leftrightarrow a) \mid a \in \{goal, at(ram, office), has(ram, cash), has(ram, icard)\}\} \cup \\ &\{\Box(L_\bot \rightarrow \bot), \\ &\Box(L_{at(ram,office) \wedge has(ram,cash)} \leftrightarrow L_{at(ram,office)} \wedge L_{has(ram,cash)}), \\ &\Box(L_g \leftrightarrow L_{at(ram,office) \wedge has(ram,cash)} \wedge L_{has(ram,icard)}), \\ &\Box(L_{g \rightarrow goal} \leftrightarrow L_g \rightarrow L_{goal}), \\ &\Box(L_{\Diamond_{\leq 60}\, goal} \leftrightarrow \Diamond_{\leq 60} L_{goal}), \\ &\Box(L_{\Diamond_{\leq 60}\, goal \rightarrow \bot} \leftrightarrow L_{\Diamond_{\leq 60}\, goal} \rightarrow L_\bot), \\ &\Box(L_q \leftrightarrow (L_{\Diamond_{\leq 60}\, goal \rightarrow \bot}) \rightarrow L_\bot)\} \end{aligned}$$

While program $\Theta(q)$ correctly captures the goal condition, it can be simplified and directly expressed as a metric logic program as follows:

$$\Box(goal \leftarrow at(ram, office) \wedge has(ram, cash) \wedge has(ram, icard)) \tag{26}$$

$$\Box(\bot \leftarrow \mathbf{I} \wedge \neg\Diamond_{\leq 60}\, goal) \tag{27}$$

To show the correspondence between the models of $\varphi$ and $\Theta(\varphi)$, we define the restriction of an HT-trace to an alphabet $\mathcal{A}$ as

$$(\langle H_i, T_i\rangle)_{0 \leq i < \lambda}|_{\mathcal{A}} = (\langle H_i \cap \mathcal{A}, T_i \cap \mathcal{A}\rangle)_{0 \leq i < \lambda} \tag{28}$$

*Theorem 5 (Correctness and Completeness).*
Let $\varphi$ be a metric formula over $\mathcal{A}$. Then, we have[6]

$$\{(\langle \mathbf{H}, \mathbf{T}\rangle, \tau) \mid (\langle \mathbf{H}, \mathbf{T}\rangle, \tau) \models \varphi\} = \{(\langle \mathbf{H}', \mathbf{T}'\rangle|_{\mathcal{A}}, \tau) \mid (\langle \mathbf{H}', \mathbf{T}'\rangle, \tau) \models \Theta(\varphi)\}$$

### *4.1 Base translation of general metric logic programs*

Analogous to Section 3, our translation of general metric logic programs is divided into two parts: a basic temporal component that addresses state transitions, and metric components that map timing functions to either HT or $\mathrm{HT}_c$. The metric components are detailed in Sections 4.2 and 4.3. In the following, we focus on the basic temporal component.

Given a metric logic program $P$ over $\mathcal{A}$ and $\lambda \in \mathbb{N}$, we define

$$\mathbf{\Pi}_\lambda(P) = \{(r)_k \mid \Box r \in P, 0 \leq k < \lambda\} \cup \bigcup\nolimits_{\varphi \in r, \Box r \in P, 0 \leq k < \lambda} \eta_k^*(\varphi) \tag{29}$$

where $(r)_k$ is obtained from $r$ by replacing in $r$ each metric atom $\mu$ by $L_\mu^k$.

The Tseitin-style translation of each metric formula in $r$ is given in the right column of Table 3. It relies on additional auxiliary atoms of form $L_\varphi^k$, $L_\varphi^{k,j}$ and $\delta_I^{k,j}$. Atoms of form

[6] We let $(\langle \mathbf{H}', \mathbf{T}'\rangle, \tau) \models \Theta(\varphi)$ abbreviate that $(\langle \mathbf{H}', \mathbf{T}'\rangle, \tau)$ is a MHT-model of $\Theta(\varphi)$.

Downloaded from https://www.cambridge.org/core. UNIVERSIDAD AUTONOMA METROPOLITANA, on 08 Nov 2026 at 14:47:57, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://doi.org/10.1017/S1471068426100714

Table 3. *Definition of sets* $\eta_k(\varphi)$ *and* $\eta_k^*(\varphi)$ *for any metric atom* $\varphi$ *and* $0 \leq k < \lambda$

| $\varphi$ | $\eta_k(\varphi)$ | $\eta_k^*(\varphi)$ |
|---|---|---|
| $a$ | $L_a^k \leftrightarrow a_k$ | $a_k \rightarrow L_a^k$<br>$L_a^k \rightarrow a_k$ |
| $\bot$ | $L_\bot^k \leftrightarrow \bot$ | $L_\bot^k \rightarrow \bot$ |
| $\top$ | $L_\top^k \leftrightarrow \top$ | $\top \rightarrow L_\top^k$ |
| $\mathbf{F}$ | $L_\mathbf{F}^k \leftrightarrow \begin{cases} \top & \text{if } k = \lambda - 1 \\ \bot & \text{if } k < \lambda - 1 \end{cases}$ | $\top \rightarrow L_\mathbf{F}^{\lambda-1}$<br>$L_\mathbf{F}^k \rightarrow \bot$ if $k < \lambda - 1$ |
| $\mathbf{I}$ | $L_\mathbf{I}^k \leftrightarrow \begin{cases} \top & \text{if } k = 0 \\ \bot & \text{if } k > 0 \end{cases}$ | $\top \rightarrow L_\mathbf{I}^0$<br>$L_\mathbf{I}^k \rightarrow \bot$ if $k > 0$ |
| $\bigcirc_I b$ | $L_{\bigcirc_I b}^k \leftrightarrow \begin{cases} L_b^{k+1} \wedge \neg\delta_I^{k,k+1} & \text{if } k < \lambda - 1 \\ \bot & \text{if } k = \lambda - 1 \end{cases}$ | $L_{\bigcirc_I b}^k \rightarrow \begin{cases} L_b^{k+1} & \text{if } k < \lambda - 1 \\ \bot & \text{if } k = \lambda - 1 \end{cases}$<br>$L_{\bigcirc_I b}^k \wedge \delta_I^{k,k+1} \rightarrow \bot$ if $k < \lambda - 1$<br>$L_b^{k+1} \wedge \neg\delta_I^{k,k+1} \rightarrow L_{\bigcirc_I b}^k$ if $k < \lambda - 1$ |
| $\Box_I b$ | $L_{\Box_I b}^k \leftrightarrow \bigwedge_{j=k}^{\lambda-1} L_{\Box_I b}^{k,j}$<br>$L_{\Box_I b}^{k,j} \leftrightarrow \delta_I^{k,j} \vee L_b^j$ for $k \leq j < \lambda$ | $\bigwedge_{j=k}^{\lambda-1} L_{\Box_I b}^{k,j} \rightarrow L_{\Box_I b}^k$<br>$L_{\Box_I b}^k \rightarrow L_{\Box_I b}^{k,j}$ for $k \leq j < \lambda$<br>$\delta_I^{k,j} \rightarrow L_{\Box_I b}^{k,j}$ for $k \leq j < \lambda$<br>$L_b^j \rightarrow L_{\Box_I b}^{k,j}$ for $k \leq j < \lambda$<br>$L_{\Box_I b}^{k,j} \wedge \neg\delta_I^{k,j} \rightarrow L_b^i$ for $k \leq j < \lambda$ |
| $\Diamond_I b$ | $L_{\Diamond_I b}^k \leftrightarrow \bigvee_{j=k}^{\lambda-1} L_{\Diamond_I b}^{k,j}$<br>$L_{\Diamond_I b}^{k,j} \leftrightarrow \neg\delta_I^{k,j} \wedge L_b^j$ for $k \leq j < \lambda$ | $L_{\Diamond_I b}^k \rightarrow \bigvee_{j=k}^{\lambda-1} L_{\Diamond_I b}^{k,j}$<br>$L_{\Diamond_I b}^{k,j} \rightarrow L_{\Diamond_I b}^k$ for $k \leq j < \lambda$<br>$L_{\Diamond_I b}^{k,j} \wedge \delta_I^{k,j} \rightarrow \bot$ for $k \leq j < \lambda$<br>$L_{\Diamond_I b}^{k,j} \rightarrow L_b^j$ for $k \leq j < \lambda$<br>$\neg\delta_I^{k,j} \wedge L_b^j \wedge \rightarrow L_{\Diamond_I b}^{k,j}$ for $k \leq j < \lambda$ |

$L_\varphi^k$ provide an equivalence to the satisfaction of $\varphi$ at state $k$. Atoms of form $L_\varphi^{k,j}$ are used to define $L_\varphi^k$ and indicate state $j$ as a witness to the truth of $\varphi$ at state $k$. Atoms of form $\delta_I^{k,j}$ provide the link to the respective encoding of metric information, defined in the next two sections. Specifically, $\delta_I^{k,j}$ indicates whether a transition between states $k$ and $j$ fails to be done within the time allotted by interval $I$, viz. $\tau(j) - \tau(k) \notin I$. Interestingly, such atoms only occur in rule bodies in $\eta_k^*(\varphi)$.

For illustration consider the metric rule in equation (27). Our translation first yields:

$$L_\bot^k \leftarrow L_\mathbf{I}^k \wedge \neg L_{\Diamond_{\leq 60}\, goal}^k \qquad \text{for } 0 \leq k < \lambda \tag{30}$$

Downloaded from https://www.cambridge.org/core. UNIVERSIDAD AUTONOMA METROPOLITANA, on 08 Nov 2026 at 14:47:57, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://doi.org/10.1017/S1471068426100714

Followed by the implications obtained from the translation of each metric formula in the rule:

$$L^k_{\bot} \leftrightarrow \bot \qquad \text{for } 0 \leq k < \lambda \tag{31}$$

$$L^0_{\mathbf{I}} \leftrightarrow \top \tag{32}$$

$$L^k_{\mathbf{I}} \leftrightarrow \bot \qquad \text{for } 0 \leq k < \lambda - 1 \tag{33}$$

$$L^k_{goal} \leftrightarrow goal_k \qquad \text{for } 0 \leq k < \lambda \tag{34}$$

$$L^k_{\Diamond_{\leq 60}\, goal} \leftrightarrow \bigvee_{j=k}^{\lambda-1} L^{k,j}_{\Diamond_I goal} \qquad \text{for } 0 \leq k < \lambda \tag{35}$$

$$L^{k,j}_{\Diamond_{\leq 60}\, goal} \leftrightarrow \neg\delta^{k,j}_I \wedge L^j_{goal} \qquad \text{for } 0 \leq k < \lambda, k \leq j < \lambda \tag{36}$$

Observe that when $\delta^{k,j}_I$ is false, the witness of truth for $\Diamond_{\leq 60}\, goal$ at state $j$ in equation (36) simplifies to $L^{k,j}_{\Diamond_{\leq 60}\, goal} \leftrightarrow L^j_{goal}$. This indicates that the metric condition over the interval is trivially satisfied, allowing any state that meets the goal condition to serve as a witness, regardless of its associated time point. Conversely, for the $\Box$ operator, such a simplification of $\delta^{k,j}_I$ would imply that all states lie within the interval. This would strengthen the witness requirements, enforcing compliance with the condition across all future states.

Clearly, our translation still bears redundancies and can be further optimized.

### *4.2 Translation of general metric logic programs to HT*

This section extends the definition of timing functions within HT from plain to general metric logic programs. Timing functions are formalized as in Section 3.1, utilizing a set of atoms $\mathcal{T} = \{t_{k,d} \mid k, d \in \mathbb{N}\}$ and the axioms of $\Delta_{\lambda,\nu}$ in equation (13). Each atom $t_{k,d}$ represents that $\tau(k) = d$. The key distinction lies in the formalization of how these timing functions interact with the interval conditions specified in the program.

To this end, let $\mathbb{I}|_P$ stand for all intervals occurring in a metric logic program $P$.

Given $\lambda, \nu \in \mathbb{N}$ and a general metric program $P$, we let

$$\begin{aligned}\mathbf{\Psi}_{\lambda,\nu}(P) = \{&\delta^{k,j}_{[m..n)} \leftarrow t_{k,d} \wedge t_{j,d'} \mid 0 \leq k \leq j < \lambda - 1, 0 \leq d < d' \leq \nu, \\ &d' - d < m, [m..n) \in \mathbb{I}|_P\} \cup \end{aligned} \tag{37}$$

$$\begin{aligned}\{&\delta^{k,j}_{[m..n)} \leftarrow t_{k,d} \wedge t_{j,d'} \mid 0 \leq k \leq j < \lambda - 1, 0 \leq d < d' \leq \nu, \\ &d' - d \geq n, [m..n) \in \mathbb{I}|_P, n \in \mathbb{N}\}\end{aligned} \tag{38}$$

Differing from the rule-oriented integrity constraints used to define $\Psi_{\lambda,\nu}(P)$ in equations (14) and (15), we now utilize rules that directly signal interval condition violations. Each such violation is indicated by an atom $\delta^{k,j}_I$, which signifies that the difference $\tau(j) - \tau(k)$ falls outside the interval $I$, where $k$ and $j$ are indices representing specific time points.

For the interval [15..16) in rule equation (10), we get

$$\delta^{k,j}_{[15..16)} \leftarrow t_{k,d} \wedge t_{j,d'} \qquad \text{for } d' - d < 15 \tag{39}$$

$$\delta^{k,j}_{[15..16)} \leftarrow t_{k,d} \wedge t_{j,d'} \qquad \text{for } d' - d \geq 16 \tag{40}$$

Downloaded from https://www.cambridge.org/core. UNIVERSIDAD AUTONOMA METROPOLITANA, on 08 Nov 2026 at 14:47:57, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://doi.org/10.1017/S1471068426100714

and $0 \leq k \leq j < \lambda - 1$, $0 \leq d < d' \leq \nu$. For $\lambda = 100$ and $\nu = 1000$, this then amounts to roughly $10^{10}$ instances for each of the above constraints. Showing an increase by a factor of $10^2$ wrt. Section 3.1 which amounts for the addition of $j$ ranging from $k$ to 100. Since $\delta_I^{k,j}$ is used to express falling outside the interval, intervals like [15..16), with a distance of 1, generate the largest number of rules. On the other hand, the interval $[0..\omega)$, used in rules equations (8) and (9), does not yield a single rule. Furthermore, the instance of rule equation (7) for moving from *atm* to *home* uses the same interval as rule equation (10), since the distance between these locations is the same. In contrast to Section 3.1, the rules for this interval condition are reused.

This approach culminates in the following completeness and correctness result.

*Theorem 6 (Completeness).*
Let $P$ be a metric logic program over $\mathcal{A}$, and $\mathbf{M} = (\langle \mathbf{T}, \mathbf{T} \rangle, \tau)$ a total timed HT-trace of length $\lambda$.

If $\mathbf{M}$ is a metric equilibrium model of $P$, then there exists an equilibrium model $\langle T, T \rangle$ of $\mathbf{\Pi}_\lambda(P) \cup \Delta_{\lambda,\nu} \cup \mathbf{\Psi}_{\lambda,\nu}(P)$ with $\nu = \tau(\lambda - 1)$ such that $\langle T, T \rangle|_{\mathcal{A}^* \cap \mathcal{T}} = \theta(\mathbf{M})$.

*Theorem 7 (Correctness).*
Let $P$ be a metric logic program, and $\lambda, \nu \in \mathbb{N}$.

If $\langle T, T \rangle$ is an equilibrium model of $\mathbf{\Pi}_\lambda(P) \cup \Delta_{\lambda,\nu} \cup \mathbf{\Psi}_{\lambda,\nu}(P)$, then $\sigma(\langle T, T \rangle|_{\mathcal{A}^* \cap \mathcal{T}})$ is a metric equilibrium model of $P$.

The restriction to $\langle T, T \rangle|_{\mathcal{A}^* \cap \mathcal{T}}$ allows us to ignore auxiliary atoms.

### *4.3 Translation of general metric logic programs to HT$_c$*

This section adapts the framework of the preceding one, utilizing HT$_c$ as the target system instead of HT, for extending the definitions of plain metric programs from Section 3.2 to general ones. Specifically, we keep the HT$_c$ signature $\langle \mathcal{X}, \mathcal{D}, \mathcal{C} \rangle$ and extend the set $\mathcal{X}$ to include Boolean variables of the form $L_\mu^k$ $L_\mu^{k,j}$ and $\delta_k^{j,I}$. The denotation for Boolean atoms, equalities, and difference constraints remains as defined in Section 3.2. Furthermore, we keep employing integer variables $\mathcal{T}^c = \{t_k \mid 0 \leq k < \lambda\}$ to directly represent timing functions, such that $\tau(k) = t_k$ for $0 \leq k < \lambda$, adhering to the axioms of $\Delta_\lambda^c$ in equation (18).

However, to formalize the interaction between these timing functions and the interval conditions specified in the program, we adopt the approach of the preceding section, deriving atoms of the form $\delta_I^{k,j}$ indicating that the difference $\tau(j) - \tau(k)$ falls outside interval $I$.

$$\mathbf{\Psi}_\lambda^c(P) = \{\delta_{[m..n)}^{k,j} \leftarrow \neg(t_k - t_j \leq -m) \mid 0 \leq k \leq j < \lambda - 1, [m..n) \in \mathbb{I}|_P\} \cup \tag{41}$$

$$\{\delta_{[m..n)}^{k,j} \leftarrow \neg(t_j - t_k \leq n - 1) \mid 0 \leq k \leq j < \lambda - 1, [m..n) \in \mathbb{I}|_P, n \in \mathbb{N}\} \tag{42}$$

For the interval [15..16) in rule equation (10), we get

$$\delta_{[15..16)}^{k,j} \leftarrow \neg(t_k - t_j \leq -15) \tag{43}$$

$$\delta_{[15..16)}^{k,j} \leftarrow \neg(t_j - t_k \leq 15) \tag{44}$$

Downloaded from https://www.cambridge.org/core. UNIVERSIDAD AUTONOMA METROPOLITANA, on 08 Nov 2026 at 14:47:57, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://doi.org/10.1017/S1471068426100714

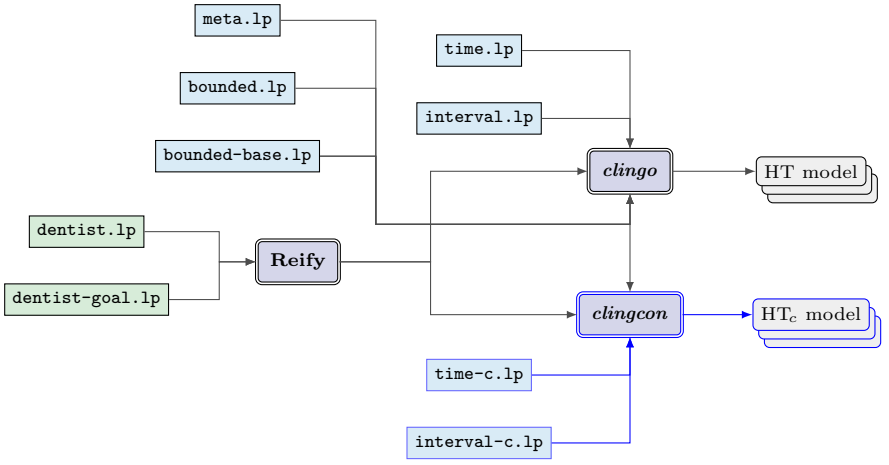



Fig. 1. Workflow of our implementations applied to the dentist example.

for $0 \leq k \leq j < \lambda - 1$. Given $\lambda = 100$, this amounts to $10^4$ instances, showing the same increase proportional $\lambda$ as in the HT approach with respect to the plain metric fragment for $\mathrm{HT}_c$, due to the inclusion of $j$.

To show completeness and correctness, we define the restriction of an $\mathrm{HT}_c$ interpretation to an alphabet $\mathcal{A}$ as $\langle h, t\rangle|_{\mathcal{A}} = \langle h|_{\mathcal{A}}, t|_{\mathcal{A}}\rangle$ where $f|_{\mathcal{A}}$ is defined such that $f|_{\mathcal{A}}(a) = f(a)$ for all $a \in \mathcal{A}$.

*Theorem 8 (Completeness).*
Let $P$ be a metric logic program over $\mathcal{A}$, and $\mathbf{M} = (\langle \mathbf{T}, \mathbf{T}\rangle, \tau)$ a total timed HT-trace of length $\lambda$.

If $\mathbf{M}$ is a metric equilibrium model of $P$, then there exists a constraint equilibrium model $\langle t, t\rangle$ of $\mathbf{\Pi}_\lambda(P) \cup \Delta^c_\lambda \cup \mathbf{\Psi}^c_\lambda(P)$ such that $\langle t, t\rangle|_{\mathcal{A}^* \cap \mathcal{T}^c} = \theta^c(\mathbf{M})$.

*Theorem 9 (Correctness).*
Let $P$ be a metric logic program, and $\lambda \in \mathbb{N}$.

If $\langle t, t\rangle$ is an constraint equilibrium model of $\mathbf{\Pi}_\lambda(P) \cup \Delta^c_\lambda \cup \mathbf{\Psi}^c_\lambda(P)$, then $\sigma^c(\langle t, t\rangle|_{\mathcal{A}^* \cap \mathcal{T}^c})$ is a metric equilibrium model of $P$.

## 5 Implementation

In what follows, we rely on a basic acquaintance with the ASP system *clingo* (Gebser *et al.* 2015). We show below how easily our approach is implemented via *clingo*'s meta-encoding framework. This serves us as a blueprint for a more sophisticated future implementation.

However, before delving into details, we give in Figure 1 an overview of our framework and its workflows. File names depicted in the diagram correspond to those introduced in the listings of this section. Nodes with a green background represent user-provided input programs, illustrated here by the dentist example, whereas solver nodes are distinguished by a purple double border. The blue nodes denote the meta-encodings used for the translation, some of which are shared by both approaches, while the $\mathrm{HT}_c$-specific

Downloaded from https://www.cambridge.org/core. UNIVERSIDAD AUTONOMA METROPOLITANA, on 08 Nov 2026 at 14:47:57, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://doi.org/10.1017/S1471068426100714

```
1  item(icard). item(cash).
2  loc(dentist). loc(office). loc(atm). loc(home).
4  at(ram,office) :- initially.
5  at(cash,atm)   :- initially.
6  at(icard,home) :- initially.

8  go(ram,L') : loc(L'), L'!=L :- at(ram,L), not finally.

10 has(ram,I) :- at(ram,L), at(I,L), item(I).
11 at(ram,L)  :- at(ram,L), has(ram,I).

13 next((D,D+1),at(ram,L')) :- at(ram,L), go(ram,L'), distance(L,L',D).

15 next((0,w),has(ram,I)) :- has(ram,I), not finally.
16 next((0,w),at(I,L)) :- at(I,L), item(I), not has(_,I), not finally.
```

Listing 1. Metric logic program for dentist example (`dentist.lp`)

pipeline is explicitly highlighted with blue arrows. All presented encodings, as well as our evaluation results, are publicly available at https://github.com/potassco/memelingo.

*Clingo* allows for reifying a ground logic program in terms of facts, which can then be (re)interpreted by a meta-encoding. The result of another grounding is then channeled to the respective back-end, in our case a regular or hybrid ASP solver, respectively. Though, for brevity, we must refer to Kaminski *et al.* (2023) for details, we mention that a reified ground program is represented by instances of predicates `atom_tuple`, `literal_tuple`, `rule`, `output`, etc.[7]

Listing 1 uses the dentist example (leaving out the representation of Table 1 in terms of `distance/3`) to illustrate a metric logic program. This input program is then reified and interpreted by the corresponding meta-encodings. For simplicity, we assume that each rule is implicitly in the scope of an always operator $\Box$. Moreover, included temporal operators and their comprised atoms are exempt from simplifications during grounding.[8] We use predicate `next/2` for the metric next operator. As an example, consider the instance of Line 13 for moving from *office* to *home*, viz. the counterpart of equation (10).

```
1  next((15,16),at(ram,home)) :-
2                at(ram,office), go(ram,home), distance(office,home,15).
```

Note that `next((0,w),.)` in the head of the last two rules stands for $\bigcirc$ aka $\bigcirc_{[0..\omega)}$.

The rules in Listing 1 do not ensure that Ram reaches the dentist within one hour with all necessary items. This condition is represented in Listing 2, corresponding to rules equations (26) and (27). However, these rules use the global metric operator $\Diamond_I$, which means they do not conform a (plain) metric logic program. As a result, we must adopt a different approach for these programs, replacing Listing 2, which will be discussed in Section 5.1. Before proceeding, we will conclude this section with the remaining implementation details that are common to both fragments.

Listing 3 modifies the basic meta-encoding in Kaminski *et al.* (2023) by adding a variable `T` for time steps to all derived predicates. Their range is fixed in Line 1. In this

[7] Below we draw upon the symbol table, captured by `output/2`, for extracting syntactic entities.
[8] In meta-encodings, this is done by adding corresponding `#external` directives.

Downloaded from https://www.cambridge.org/core. UNIVERSIDAD AUTONOMA METROPOLITANA, on 08 Nov 2026 at 14:47:57, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://doi.org/10.1017/S1471068426100714

```
1  goal :- at(ram,dentist), has(ram,icard), has(ram,cash).
2  :- initially, not eventually((0,60),goal).
```

Listing 2. Metric logic program for the goal in dentist example (`dentist-goal.lp`)

```
1  time(0..lambda-1).

3  conjunction(B,T) :- literal_tuple(B), time(T),
4         hold(L,T) : literal_tuple(B, L), L > 0;
5     not hold(L,T) : literal_tuple(B,-L), L > 0.

7  body(normal(B),T) :- rule(_,normal(B)), conjunction(B,T).
8  body(sum(B,G),T)  :- rule(_,sum(B,G)), time(T),
9    #sum{W,L :      hold(L,T), weighted_literal_tuple(B, L,W), L>0;
10        W,L : not hold(L,T), weighted_literal_tuple(B,-L,W), L>0} >= G.

12   hold(A,T) : atom_tuple(H,A)   :- rule(disjunction(H),B), body(B,T).
13 { hold(A,T) : atom_tuple(H,A) } :- rule(
   choice(H),B), body(B,T).
```

Listing 3. Timed meta-encoding (`meta.lp`)

```
1  true(O,K) :- output(O,B), time(K), hold(L,K) : literal_tuple(B,L).
2  hold(L,K) :- output(O,B), time(K), true(O,K),
   literal_tuple(B,L).

4  :- true(initially,K), time(K), K!=0.
5  true(initially,0).

7  :- true(finally,K), time(K), K!=lambda-1.
8  true(finally,lambda-1).
```

Listing 4. Meta encoding for base metric operators (`bounded-base.lp`)

way, an atom `hold(a,k)` stands for $a_k$ in $\mathcal{A}^*$, where `a` is the numeric identifier of $a$ in $\mathcal{A}$. While this encoding handles Boolean connectives, the metric ones are treated in Listing 4. The rules in Line 1 and 2 restore the symbolic representation of the numerically identified atoms, which allows us to analyze the inner structure of modalized propositions. Lines (4/5) and (7/8) deal with $\mathbf{I}$ and $\mathbf{F}$, respectively. The remaining metric operators are handled differently for each metric fragment, as described in Sections 3 and 4. These will be discussed in detail in Sections 5.1 and 5.2, respectively.

### *5.1 Implementation of plain metric logic programs*

Lines 1 and 2 of Listing 5 realize the metric next operator, $\bigcirc_I a$, represented by term `next(I,a)`. Together, Listing 3, 4 and 5 account for $\Pi_\lambda(P)$.

When expressing time via Boolean variables, the previous listings are combined with Listing 6 and 7 below, which realize $\Delta_{\lambda,\nu}$ and $\Psi_{\lambda,\nu}(P)$, respectively. Atoms $t_{k,d}$ in $\mathcal{T}$ are represented by `t(k,d)`. The upper bound $\nu$ on the timing function's range is given by `v`, and $\omega$ is represented by `w`. The two encodings directly mirror the definitions of $\Delta_{\lambda,\nu}$ and

Downloaded from https://www.cambridge.org/core. UNIVERSIDAD AUTONOMA METROPOLITANA, on 08 Nov 2026 at 14:47:57, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://doi.org/10.1017/S1471068426100714

```
1  :- true(next(_,_),K), time(K), K=lambda-1.
2  true(V,K+1) :- true(next(I,V),K).
```

Listing 5. Meta encoding for metric operators of $\Pi_\lambda(P)$ (`bounded.lp`)

```
1  timepoint(0..v).

3  t(0,0).
4  t(K+1,D') : D<D', timepoint(D') :- t(K,D), time(K+1).
```

Listing 6. Meta encoding for $\Delta_{\lambda,\nu}$ (`time.lp`)

```
1  :- true(next((M,N),V),K), t(K,D), t(K+1,D'), D' - D < M.
2  :- true(next((M,N),V),K), t(K,D), t(K+1,D'), D' - D >= N , N!=w.
```

Listing 7. Meta encoding for $\Psi_{\lambda,\nu}(P)$ (`interval.lp`)

```
1  &sum{t(0)} = 0.
2  &sum{t(K) ; -t(K+1)} <= -1 :- time(K), time(K+1).
```

Listing 8. Meta encoding for $\Delta^c_\lambda$ (`time-c.lp`)

```
1  &sum{t(K); -t(K+1)} <= -M   :- true(next((M,N),V),K),
2                                 time(K), time(K+1).
3  &sum{t(K+1); -t(K)} <= N-1 :- true(next((M,N),V),K),
4                                 time(K), time(K+1), N!=w.
```

Listing 9. Meta encoding for $\Psi^c_\lambda(P)$ (`interval-c.lp`)

$\Psi_{\lambda,\nu}(P)$, with one key difference: the rule bodies in equations (14) and (15) are replaced in Lines 1 and 2 of Listing 7 by auxiliary atoms of predicate `true/2`.

When expressing time in terms of integer variables, we rely on difference constraints for modeling timing functions. Such simplified linear constraints have the form "$x - y \leq d$" for $x, y \in \mathcal{X}$ and $d \in \mathbb{Z}$ and are supported by the *clingo* extensions *clingcon* (Banbara *et al.* 2017) and *clingo*[DL] (Janhunen *et al.* 2017). We use below *clingcon*'s syntax and represent them as "`&sum x ;` $y$ `<=` $d$." A Boolean atom $a$ can be seen as representing "$a = \boldsymbol{t}$." In the case at hand, Listing 3, 4 and 5 are now completed by Listing 8 and 9 below. As above, they faithfully replicate the definitions of $\Delta^c_\lambda$ and $\Psi^c_\lambda(P)$. Unlike above, however, the timing function is now captured by integer variables of form `t(k)` and its range restriction is now obsolete. The rules in Listing 8 mirror the two conditions on timing functions, namely, that `t(0)` equals zero and that the instances of `t(K+1)` receive a strictly greater integer than the ones of `t(K)` for `K` ranging from `0` to `lambda-1`. Similarly, given that `w` stands for $\omega$, the two rules in Listing 9 correspond to the difference constraints in equations (19) and (20).

As in Listing 7, we use auxiliary atoms of predicate `true/2` rather than the corresponding rule bodies. Notably, we shifted in Listing 9 the difference constraints in equations (19) and (20) from the body to the head. This preserves (strong) equivalence in $\mathrm{HT}_c$ whenever all variables comprised in a constraint atom are defined, as guaranteed by Proposition 3.

Downloaded from https://www.cambridge.org/core. UNIVERSIDAD AUTONOMA METROPOLITANA, on 08 Nov 2026 at 14:47:57, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://doi.org/10.1017/S1471068426100714

In the two approaches at hand, we may compensate the lack of global metric operators in out example by replacing Line 1 by `:- finally, not goal` and enforcing the time limit either by setting $\nu$ to 60 in our HT-based approach or by extending Listing 5 with `&sumt(K) <= 60 :- time(K), not time(K+1).` in our HT$_c$-based approach. However, this is only effective when the goal is achieved at the final step.

The example in Listing 1 is addressed with *clingo* in the following way.

```
clingo dentist.lp --output=reify |
clingo 0 - meta.lp bounded-base.lp bounded.lp
           time.lp interval.lp -c lambda=4 -c v=110
```

When using *clingcon* instead, it suffices to replace the second line with:

```
clingcon 0 - meta.lp bounded-base.lp bounded.lp
             time-c.lp interval-c.lp -c lambda=4
```

Our choices of `lambda` and `v` allow for all movement combinations within the 4 steps required to reach the goal; we obtain in each case 27 solutions. Once we include the query-oriented additions from above, we obtain a single model instead.

### *5.2 Implementation of general metric logic programs*

Listing 10 extends Listing 4 to account for the metric next operator $\bigcirc_I b$ in Lines 1 to 5, the metric always operator $\Box_I b$ in Lines 7 to 16, and the metric eventually operator $\Diamond_I b$ in Lines 18 to 25. These rules mirror the definitions of $\eta_k^*(\varphi)$ in Table 3,[9] where atoms `true(m,k)` correspond to $L_\varphi^k$, witnesses $L_\varphi^{k,j}$ are expressed via the atom `wit(m,(k,j))`, and interval violations $\delta_i^{k,j}$ are represented by `iv(i,(k,j))`. Together, Listings 3, 4, and 10 account for $\mathbf{\Pi}_\lambda(P)$.

When expressing time via Boolean variables, we reuse Listing 6 to implement $\Delta_{\lambda,\nu}$ and define $\mathbf{\Psi}_{\lambda,\nu}(P)$ in Listing 11 below. The two rules in Lines (1/2) and Lines (3/4) correspond to equation (37) and (38), respectively, where predicate `i/1` collects the intervals in $\mathbb{I}|_P$

When expressing time using integer variables, we rely on difference constraints to model timing functions, as described in the previous section. We reuse Listing 8 to implement $\Delta_\lambda^c$ and define $\mathbf{\Psi}_\lambda^c(P)$ in Listing 12 below. The two rules in Listing 12 correspond to rules equations (41) and (42).

The example in Listing 1 can be addressed using *clingo* and *clingcon* in the same manner as before. However, this fragment accommodates the goal condition from Listing 2. Listing 13 displays the corresponding *clingo* output. For readability, we ordered the output atoms and omitted all predicates except `t/2`, `go/3`, and `at/3`. The output presents the expected optimal plan for reaching the dentist on time. Line 6 details Ram's actions (traveling from the office to the ATM, then home, and finally to the dentist), while Lines 7–9 track his location and the items he acquires. The timing function is encoded as normal atoms over the predicate `t/2` (Line 11); the time difference between consecutive states corresponds to the distance between locations, achieving the goal in 55 minutes.

[9] The use of the predicate `output/2` is merely to ensure rule safety.

Downloaded from https://www.cambridge.org/core. UNIVERSIDAD AUTONOMA METROPOLITANA, on 08 Nov 2026 at 14:47:57, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://doi.org/10.1017/S1471068426100714

```
true(V,K+1) :- true(next(I,V),K), time(K), K<lambda-1.
:- true(next(_,_),K), time(K), K=lambda-1.
:- true(next(I,_),K), iv(I,(K,K+1)), time(K), K<lambda-1.
true(next(I,V),K) :-  true(V,K+1), not iv(I,(K,K+1)),
                      time(K), K<lambda-1, output(next(I,V),_).

true(always(I,V),K) :- wit(always(I,V),(K,J)) : time(J), K<=J;
                       output(always(I,V),_), time(K).
wit(always(I,V),(K,J)) :- true(always(I,V),K),
                          time(K), time(J), K<=J.
wit(always(I,V),(K,J)) :- iv(I,(K,J)), output(always(I,V),_),
                          time(K), time(J), K<=J.
wit(always(I,V),(K,J)) :- true(V,J), output(always(I,V),_),
                          time(K), time(J), K<=J.
true(V,J) :- wit(always(I,V),(K,J)), not iv(I,(K,J)),
             time(K), time(J), K<=J.

wit(eventually(I,V),(K,J)): time(J), K<=J :- true(eventually(I,V),K).
true(eventually(I,V),K) :- wit(eventually(I,V),(K,J)),
                           time(K), time(J), K<=J.
                         :- wit(eventually(I,V),(K,J)), iv(I,(K,J)),
                           time(K), time(J), K<=J.
true(V,J):-wit(eventually(I,V),(K,J)),
           time(K), time(J), K<=J.
wit(eventually(I,V),(K,J)):- true(V,J), not iv(I,(K,J)),
                             output(eventually(I,V),_),
                             time(K), time(J), K<=J.
```

Listing 10. Meta encoding for metric operators of $\mathbf{\Pi}_\lambda(P)$ (`bounded.lp`)

```
iv((M,N),(K,J)):- i((M,N)), t(K,D), t(J,D'), time(K), time(J), K<=J,
                  D' - D <  M.
iv((M,N),(K,J)):- i((M,N)), t(K,D), t(J,D'), time(K), time(J), K<=J,
                  D' - D >= N , N!=w.
```

Listing 11. Meta encoding for $\mathbf{\Psi}_{\lambda,\nu}(P)$ (`interval.lp`)

```
iv((M,N),(K,K')):- i((M,N)), time(K), time(K'), K<=K',
                   &sum{t(K); -t(K')} > -M.
iv((M,N),(K,K')):- i((M,N)), time(K), time(K'), K<=K',
                   &sum{t(K'); -t(K)} > N-1, N!=w.
```

Listing 12. Meta encoding for $\mathbf{\Psi}^{\mathrm{c}}_{\lambda}(P)$ (`interval-c.lp`)

The *clingcon* output (Listing 14) differs only in its representation of the timing function, which utilizes integer variables and is presented separately as assignments in Lines 11 and 12.

### *5.3 Results*

In this section, we present some empirical results on the scalability of both approaches. We use *clingo* 5.8.0 (Kaminski *et al.* 2023) for HT, and *clingcon* 5.2.1 (Banbara *et al.* 2017) and *clingo*[DL] 1.5.0 (Janhunen *et al.* 2017) for $\mathrm{HT_c}$. In our setting, the solutions

Downloaded from https://www.cambridge.org/core. UNIVERSIDAD AUTONOMA METROPOLITANA, on 08 Nov 2026 at 14:47:57, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://doi.org/10.1017/S1471068426100714

```
UNIX > clingo dentist.lp dentist-goal.lp --output=reify |  \
       clingo 0 - meta.lp bounded-base.lp bounded.lp       \
                  time.lp interval.lp -c lambda=4 -c v=110
Solving...
Answer: 1
true(go(ram,atm),0) true(go(ram,home),1) true(go(ram,dentist),2)
true(at(ram,office),0) true(at(icard,home),0) true(at(cash,atm),0)
true(at(ram,atm),1) true(at(icard,home),1) true(at(cash,atm),1)
true(at(ram,home),2) true(at(icard,home),2)
true(at(ram,dentist),3)
t(0,0) t(1,20) t(2,35) t(3,55)
SATISFIABLE

Models       : 1
Calls        : 1
Time         : 0.332s (Solving: 0.07s 1st Model: 0.01s Unsat: 0.06s)
CPU Time     : 0.322s
```

Listing 13. System output showcasing the dentist example using *clingo*

```
UNIX > clingo dentist.lp dentist-goal.lp --output=reify | \
       clingcon 0 - meta.lp bounded-base.lp bounded.lp    \
                  time-c.lp interval-c.lp -c lambda=4
Solving...
Answer: 1
true(go(ram,atm),0) true(go(ram,home),1) true(go(ram,dentist),2)
true(at(ram,office),0) true(at(icard,home),0) true(at(cash,atm),0)
true(at(ram,atm),1) true(at(icard,home),1) true(at(cash,atm),1)
true(at(ram,home),2) true(at(icard,home),2)
true(at(ram,dentist),3)
Assignment:
t(0)=0 t(1)=20 t(2)=35 t(3)=55
SATISFIABLE

Models       : 1
Calls        : 1
Time         : 0.016s (Solving: 0.00s 1st Model: 0.00s Unsat: 0.00s)
CPU Time     : 0.013s
```

Listing 14. System output showcasing the dentist example using *clingcon*

computed by *clingcon* coincide with those defined by $\mathrm{HT}_c$, while *clingo*[DL] returns only assignments where the time points take the smallest possible positive integer values.

*Experimental setup.* We evaluated our implementation on three problem domains. For each of them, we developed a novel metric encoding. The first is the dentist scenario, used to compare both metric fragments, plain ($P$) and general ($G$), and to study the scalability of each approach and the overhead introduced by the general fragment. In this domain, we enumerate all models to ensure full inspection of the search space, with and without the goal condition. Since the formalization uses only intervals of size one, the minimality condition of *clingo*[DL] does not reduce the number of models. The second domain is Multi-Agent Path Finding (Stern *et al.* 2019). It is evaluated only for the general fragment, because the representation requires global temporal operators ($\Diamond_I, \Box_I$)

Downloaded from https://www.cambridge.org/core. UNIVERSIDAD AUTONOMA METROPOLITANA, on 08 Nov 2026 at 14:47:57, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://doi.org/10.1017/S1471068426100714

Table 4. *Results for clingo, clingcon and clingo[DL] without goal condition (times in seconds; – represents a memout)*

| | | *clingo* | | | *clingcon* | | | *clingo*[DL] | | |
|---|---|---|---|---|---|---|---|---|---|---|
| $f$ | | Solve | Ground | #Rules | Solve | Ground | #Rules | Solve | Ground | #Rules |
| 1 | $P$ | 0.03 | 1.07 | 297,383 | 0.01 | 0.14 | 2337 | 0.01 | 0.14 | 2337 |
| | $G$ | 0.29 | 0.61 | 334,458 | 0.01 | 0.22 | 2572 | 0.01 | 0.16 | 2572 |
| 4 | $P$ | 1.65 | 21.61 | 4,812,563 | 0.01 | 0.18 | 2337 | 0.01 | 0.14 | 2337 |
| | $G$ | 36.37 | 15.80 | 5,395,908 | 0.01 | 0.19 | 2572 | 0.01 | 0.19 | 2572 |
| 7 | $P$ | 16.58 | 79.35 | 14,772,743 | 0.01 | 0.14 | 2337 | 0.01 | 0.13 | 2337 |
| | $G$ | 81.56 | 67.55 | 16,555,758 | 0.02 | 0.21 | 2572 | 0.01 | 0.16 | 2572 |
| 10 | $P$ | | – | | 0.01 | 0.15 | 2337 | 0.01 | 0.18 | 2337 |
| | $G$ | | – | | 0.02 | 0.19 | 2572 | 0.01 | 0.23 | 2572 |

to express multi-agent coordination and goal conditions, which are not available in the plain fragment. In this domain, we analyze the scalability of the general fragment across all approaches in a more complex setting. We compute a single model, as enumeration would lead to a very large number of models, so the minimality imposed by *clingo*[DL] only affects which model is returned. The metric representation is based on the ASP encoding from Behrens *et al.* (2024). The third domain is Job-Shop Scheduling. It is evaluated in the general fragment, computing a single model. Here, we analyze the impact of varying the horizon on the overall scalability of each approach. The metric encoding is inspired by Hustadt *et al.* (2020).

We conducted all experiments on a cluster[10] with Intel Xeon E5-2650v4@2.9 GHz CPUs and 64 GB of memory, running Debian Linux 10, imposing per instance a timeout of 20 minutes and a memory limit of 20 GB.

### *5.3.1 Dentist scenario*

In our first setting, we use the running example from Listing 1. As explained before, we select `lambda` $= 4$ and `v` $= 110$ to generate all movement combinations within the 4 steps required to reach the goal, resulting in 27 solutions. To investigate scalability, we multiply both the durations in Table 1 and the time limit `v` by a factor $f \in \{1, 4, 7, 10\}$. The results are summarized in Table 4. They show that *clingo* does not scale well. This is most clear at $f = 10$, where it reaches the memory limit while grounding. In contrast, the performance of *clingcon* and *clingo*[DL] is independent of the time granularity, for both fragments $P$ and $G$. The inner workings of both systems differ, but in this simple example their behavior is essentially the same. As anticipated in Sections 4.2 and 4.3, there is a linear increase in the number of rules of $G$ with respect to $P$. The additional rules enable the next operator $\bigcirc_I$ in the body and define auxiliary atoms representing interval violations. These additions may explain why the general fragment relies more on solving than the plain one, as observed in Table 4. Note that the global metric operators

[10] https://www.cs.uni-potsdam.de/bs/research/labs.html#hardware

Downloaded from https://www.cambridge.org/core. UNIVERSIDAD AUTONOMA METROPOLITANA, on 08 Nov 2026 at 14:47:57, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://doi.org/10.1017/S1471068426100714

Table 5. *Results for clingo, clingcon and clingo[DL] with goal condition, (times in seconds; – indicates the memory limit was reached)*

| | | *clingo* | | | *clingcon* | | | *clingo*[DL] | | |
|---|---|---|---|---|---|---|---|---|---|---|
| $f$ | | Solve | Ground | #Rules | Solve | Ground | #Rules | Solve | Ground | #Rules |
| 1 | $G$ | 0.10 | 0.73 | 377,301 | 0.01 | 0.19 | 2731 | 0.01 | 0.20 | 2731 |
| 4 | $G$ | 28.33 | 15.69 | 6,092,421 | 0.01 | 0.21 | 2731 | 0.01 | 0.19 | 2731 |
| 7 | $G$ | 17.20 | 77.04 | 18,694,341 | 0.01 | 0.18 | 2731 | 0.01 | 0.21 | 2731 |
| 10 | $G$ | | – | | 0.01 | 0.23 | 2731 | 0.01 | 0.19 | 2731 |

$\Diamond_I$ and $\Box_I$ are not used in this setting, and consequently the intelligent grounder does not produce the corresponding ground rules.

In our second setting, we include the goal constraint from Listing 2, which can only be handled by the general fragment. We use the same parameters as before. In particular, we enumerate all models, but in this case there is always a single one. The results are summarized in Table 5. As before, *clingo* does not scale well, while the performance of *clingcon* and *clingo*[DL] is independent of the time granularity. The number of rules increases as expected, due to the goal constraint and the rules required to account for the metric eventually $\Diamond_I$ occurring in it. For *clingo* the inclusion of the goal condition leads to a slight increase in grounding time, while solving time decreases since the goal constraint significantly reduces the number of models.

### *5.3.2 Multi-agent path finding*

Multi-Agent Path Finding (MAPF, Stern *et al.* (2019)) is the problem of finding a plan for a set of agents that move from their start to goal positions. The agents move on a map represented as a graph $(\mathcal{V}, \mathcal{E})$ where vertices in $\mathcal{V}$ represent locations and edges in $\mathcal{E}$ represent movements between locations, each with a duration. A valid plan is a sequence of movements for each agent, free of vertex conflicts and swap conflicts. Vertex conflicts occur when two agents occupy the same vertex at the same time, and swap conflicts occur when two agents traverse the same edge in opposite directions at the same time.

Our formalization adapts the ASP encoding of (Behrens *et al.* (2024), Listing 5) to the metric logic program in equations (45)–(53), capturing movement durations by the timing function. We use letters $U$ and $V$ to denote vertices from $\mathcal{V}$, and if $(U, V) \in \mathcal{E}$, the duration of the movement is given by $\delta(U, V)$. We use letters $A$ and $B$ to denote agents, and each agent $A$ has a *start* vertex $S_A \in \mathcal{V}$ and a *goal* vertex $G_A \in \mathcal{V}$.

Rule equation (45) captures the initial position of each agent. Rule equation (46) chooses a possible movement along an edge $(U, V)$. Rule equation (47) captures the effect of moving from $U$ to $V$, which takes exactly $\delta(U, V)$ time units. We use the eventually operator rather than the next operator so that different agents may move asynchronously. Rule equation (48) captures inertia: an agent remains at its current vertex $U$ if it does not move along any edge $(U, V)$. Following (Behrens *et al.* 2024), an agent is at no vertex while moving, but reappears at the target vertex $V$ upon arrival. Rule equation (49) ensures that a movement is only possible if the agent is currently at the source vertex $U$. This also

Downloaded from https://www.cambridge.org/core. UNIVERSIDAD AUTONOMA METROPOLITANA, on 08 Nov 2026 at 14:47:57, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://doi.org/10.1017/S1471068426100714

```
1                        at(A,SA) :- start(A,SA), initially.
2  move(A,U,V) , not move(A,U,V) :- not finally, agent(A), edge(U,V,D).
3    eventually((D,D+1),at(A,V)) :- move(A,U,V), edge(U,V,D).
4             next((0,w),at(A,U)) :- not finally, at(A,U),
5                                    not move(A,U,V) : edge(U,V,D).

7  :- move(A,U,V), not at(A,U).
8  :- move(A,U,V), move(A,U',V'), (U,V) != (U',V').
9  :- at(A,U), at(B,U), A != B.
10 :- move(A,U,V), eventually((0,D), move(B,V,U)), edge(U,V,D).
11 :- goal(A,U), not at(A,U), finally.
```

Listing 15. Metric logic program for *MAPF*

prevents another action from being taken until the current movement is completed. Rule equation (50) enforces that each agent moves at most once per time point, given that $(U, V)$ and $(U', V')$ are distinct edges. Rule equation (51) enforces the absence of vertex conflicts, given that $A \neq B$. Rule equation (52) enforces the absence of swap conflicts using the eventually operator to ensure that no swap conflict arises during the traversal of an edge. Finally, Rule equation (53) enforces the goal condition: every agent must be at its goal vertex at the end of the plan.

$$\Box(at(A, S_A) \leftarrow \mathbf{I}) \tag{45}$$

$$\Box(move(A, U, V) \vee \neg move(A, U, V) \leftarrow \neg\mathbf{F}) \qquad (U, V) \in \mathcal{E} \tag{46}$$

$$\Box(\Diamond_{[\delta(U,V)..\delta(U,V)+1)}\, at(A, V) \leftarrow move(A, U, V)) \tag{47}$$

$$\Box(\bigcirc_{[0..\omega)}\, at(A, U) \leftarrow \neg\mathbf{F} \wedge at(A, U) \wedge \bigwedge_{(U,V)\in\mathcal{E}} \neg move(A, U, V)) \tag{48}$$

$$\Box(\bot \leftarrow move(A, U, V) \wedge \neg at(A, U)) \tag{49}$$

$$\Box(\bot \leftarrow move(A, U, V) \wedge move(A, U', V')) \quad (U, V) \neq (U', V') \tag{50}$$

$$\Box(\bot \leftarrow at(A, U) \wedge at(B, U)) \qquad A \neq B \tag{51}$$

$$\Box(\bot \leftarrow move(A, U, V) \wedge \Diamond_{[0..\delta(U,V))}\, move(B, V, U)) \tag{52}$$

$$\Box(\bot \leftarrow goal(A, U) \wedge \neg at(A, U) \wedge \mathbf{F}) \tag{53}$$

*Experiments.* We conducted experiments in two settings using the encoding in Listing 15, which is a direct mapping of the rules above. In each case, we computed one solution.

A preliminary evaluation showed that *clingcon* was not able to find a solution without a time-point limit. The reason is that the MAPF encoding allows all agents to remain idle, resulting in an unbounded timing function $\tau$. This is not the case in the dentist scenario, where at each state there must be a movement that constrains $\tau$. This poses a challenge for *clingcon*, which attempts to assign times to states without a bound, and stalls during solving. In contrast, *clingo*[DL] selects the smallest possible positive values, naturally bounding the search space. To address this issue while ensuring a fair comparison across all approaches, we impose the time-point limit $\nu$ (used in the HT approach) also for *clingcon* and *clingo*[DL] in the $\mathrm{HT}_c$ approach, by adding a numerical constraint enforcing assignments to be smaller than $\nu$.

Downloaded from https://www.cambridge.org/core. UNIVERSIDAD AUTONOMA METROPOLITANA, on 08 Nov 2026 at 14:47:57, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://doi.org/10.1017/S1471068426100714

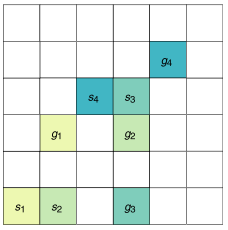



Fig. 2. Empty grid ($6 \times 6$).

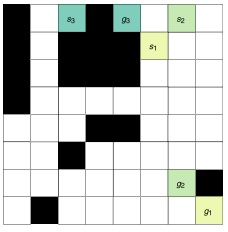



Fig. 3. Random grid ($8 \times 8$).

In the first setting, we study the scalability of the general fragment and test the time-granularity independence of the $\text{HT}_c$ approach in a more complex scenario. We use an empty grid map of size $6 \times 6$ (Figure 2) generated by the framework of Behrens *et al.* (2024). We connect vertices with edge durations ranging from 1 to 5 time units, vary the number of agents from 1 to 4, and scale edge durations by a factor $f \in \{1, 5, 10, 15, 20, 25, 30, 35\}$, solving all combinations with $\lambda = 10$ and $\nu = 15 \cdot f$. Figure 6 shows the performance of each approach across the different scaling factors. The reported time is the total time to find the first solution, and the shaded area below each line indicates the grounding time. The HT approach using *clingo* exhibits behavior similar to before: larger edge durations lead to higher runtimes and early timeouts, as well as higher grounding times, visible in the light green shaded area for the single-agent case but not for larger instances due to timeouts. In contrast, the $\text{HT}_c$ approaches with *clingcon* and *clingo*[DL] show stable performance across all scaling factors, confirming their independence from time granularity. Their grounding time is negligible across all agent counts, but the total time increases with the number of agents, as expected. Overall, *clingo*[DL] performs better than *clingcon*. Figure 7 shows the number of ground rules generated by each approach. The number of rules for *clingo* grows rapidly as the factor increases, while for *clingcon* and *clingo*[DL] the number of rules remains independent of the time granularity, growing only slightly with the number of agents due to the additional rules introduced per agent. The instances where *clingo* timed out are also plotted, as the timeout was reached during solving rather than grounding.

In the second setting, we study the effect of relaxing the horizon on solver performance across different map structures. We generated three MAPF instances on $8 \times 8$ grids, each with a different map layout: random, room, and warehouse (Figures 3, 4, and 5, respectively), fixing the number of agents to three, and sampling edge durations between 1 and 5 time units. We set a time-point limit of $\nu = 100$ and vary the horizon $\lambda$ from 15 to

Downloaded from https://www.cambridge.org/core. UNIVERSIDAD AUTONOMA METROPOLITANA, on 08 Nov 2026 at 14:47:57, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://doi.org/10.1017/S1471068426100714

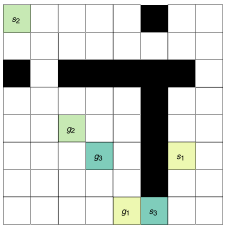



Fig. 4. Room grid ($8 \times 8$).

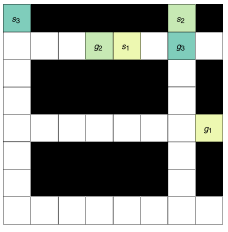



Fig. 5. Warehouse grid ($8 \times 8$).

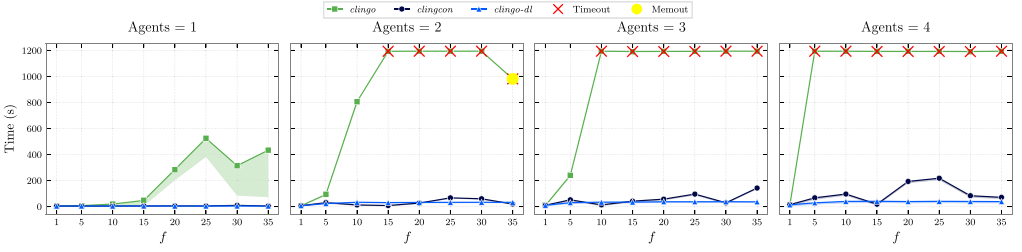



Fig. 6. Total times for empty $6 \times 6$ grid MAPF instances with increasing number of agents. The shaded areas under the lines represent grounding time.

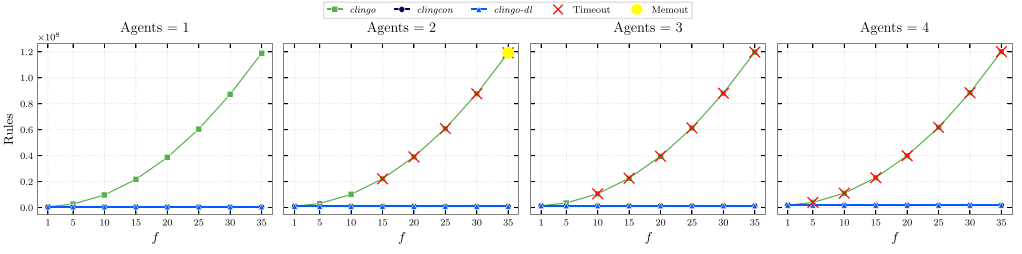



Fig. 7. Number of ground rules for empty $6 \times 6$ grid MAPF instances with increasing number of agents.

45 in steps of 5. The results are shown in Figure 8. Compared to the previous setting, grounding has a more pronounced impact, reflecting the larger map size and higher time-point limit. *Clingo*[DL] achieves the best performance across all three layouts. The behavior of *clingcon* is more irregular across layouts and values of $\lambda$, and requires further investigation. In general, as $\lambda$ increases, solving becomes faster because the problem

Downloaded from https://www.cambridge.org/core. UNIVERSIDAD AUTONOMA METROPOLITANA, on 08 Nov 2026 at 14:47:57, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://doi.org/10.1017/S1471068426100714

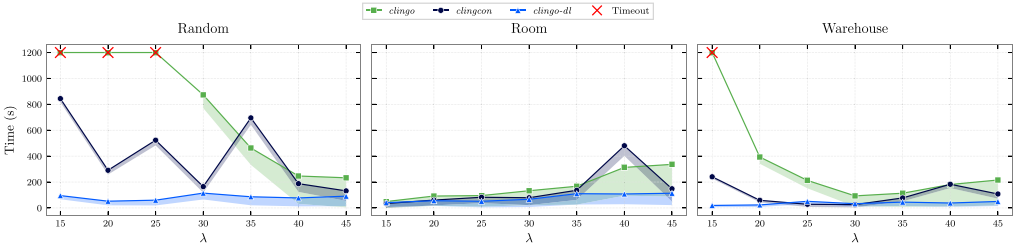



Fig. 8. Total times for $8 \times 8$ grid MAPF instances with increasing horizon. The shaded areas under the lines represent grounding time.

becomes less constrained. However, *clingo*'s grounding time grows with $\lambda$, creating a tradeoff that yields an optimal operating point, beyond which total time increases again. This underscores the importance of selecting an appropriate horizon $\lambda$.

### *5.3.3 Job-shop scheduling*

In the Job-Shop Scheduling problem, there is a set of jobs $\mathcal{J}$, operations $\mathcal{O}$ and machines $\mathcal{M}$. Each job $J \in \mathcal{J}$ consists of a sequence of operations. Each operation $O \in \mathcal{O}$ belongs to a single job and must be processed on machine $\mu(O)$ for a duration $\delta(O)$. By $prev(O)$ we denote the operation that precedes $O \in \mathcal{O}$ in the sequence, if it exists. The goal is to find a schedule such that all operations are completed in order within the given time.

Our formalization as a general metric logic program is shown in equations (54)–(59). For some operation $O$, the atom $start_run_O$ represents its start, $run_O$ represents that it is running, $has_run_O$ that it has already run, and $wait_O$ that it is waiting. Rule equation (54) captures the non-deterministic choice to start an operation, provided it has not yet completed. Rule equation (55) represents the execution of an operation for its duration, and Rule equation (56) captures its completion after that duration. Rule equation (57) prevents two distinct operations from running on the same machine at the same time, and Rule equation (58) prevents an operation from starting if its preceding operation in the job has not yet completed. Finally, Rule equation (59) enforces that all operations complete by the end of the trace.

$$\Box(wait(O) \vee start_run_O \leftarrow \neg has_run_O) \tag{54}$$

$$\Box(\Box_{[0..\delta(O))} run_O \leftarrow start_run_O) \tag{55}$$

$$\Box(\Box_{[\delta(O)..\omega)} has_run_O \leftarrow start_run_O) \tag{56}$$

$$\Box(\bot \leftarrow run_O \wedge run_{O'}) \qquad O \neq O' \text{ and } \mu(O) = \mu(O') \tag{57}$$

$$\Box(\bot \leftarrow start_run_O \wedge \neg has_run_{prev(O)}) \tag{58}$$

$$\Box(\bot \leftarrow \neg has_run_O \wedge \mathbf{F}) \tag{59}$$

*Experiments.* We study the impact of varying the horizon on the overall scalability of each approach. We evaluate the benchmark instance *ft06* from Muth and Thompson (1963), which consists of 6 jobs, 6 machines, and 36 operations. The experiments were performed with the known optimal makespan of 55, and in a relaxed setting with a time-point limit of 110 (double the optimal makespan), to analyze the impact of enforcing

Downloaded from https://www.cambridge.org/core. UNIVERSIDAD AUTONOMA METROPOLITANA, on 08 Nov 2026 at 14:47:57, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://doi.org/10.1017/S1471068426100714

```
1    wait(O) , start_run(O) :- not has_run(O), operation(O).
2      always((O,D),run(O)) :- start_run(O), duration(O,D).
3  always((D,w),has_run(O)) :- start_run(O), duration(O,D).

5  :- run(O), run(O'), O'!=O, machine(O,M), machine(O',M).
6  :- start_run(O), not has_run(O-1), job(O,J), job(O-1,J).
7  :- not has_run(O), finally, operation(O).
```

Listing 16. Metric logic program for Job-Shop Scheduling

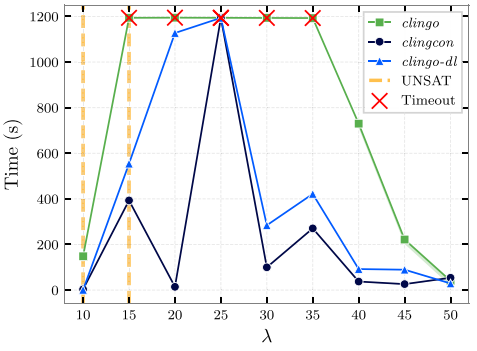



Fig. 9. Total times for *ft06* with time-point limit $\nu = 55$.

optimality via the time-point limit. The benchmark instance is translated into ASP facts using predicates `operation/1` for $O$, `machine/2` for $\mu$, `duration/2` for $\delta$, and `job/2` for the job assignment. The order of operations is captured by incremental numerical identifiers. The encoding, shown in Listing 16, is a direct mapping of Rules equations (54)–(59).

Figure 9 shows the performance of each approach across different values of $\lambda$, with the time-point limit $\nu$ fixed to the known optimal makespan of 55. The problem is unsatisfiable for $\lambda \in \{10, 15\}$; with the optimal time-point limit of 55, a horizon of $\lambda > 15$ is required to obtain a solution. *Clingo* times out for $\lambda \in \{20, 25, 30\}$, while *clingcon* and *clingo*[DL] time out at $\lambda = 25$. For all other satisfiable instances, *clingcon* and *clingo*[DL] are significantly faster than *clingo*. Unlike in the previous applications, *clingcon* shows overall better performance than *clingo*[DL] in this setting, though this warrants further analysis of the constraint solvers' internal behavior. The solving time for *clingo* generally decreases as $\lambda$ increases beyond 35, which can be attributed to the fact that a smaller horizon makes the problem harder to solve, similarly to the MAPF results. The fast response observed at $\lambda = 50$ for all approaches can be attributed to the fact that, with a time-point limit of 55, this horizon is very close to an identity mapping for $\tau$, which simplifies the search considerably. The grounding time is negligible for all approaches, as this job-shop instance is designed to be hard to solve but not to ground.

Figure 10 shows the results with a relaxed time-point limit of 110. Relaxing the time-point limit eliminates most of the timeouts observed for *clingo* and removes the erratic behavior seen in Figure 9, with only $\lambda = 10$ remaining unsolvable within the time limit. The $\mathrm{HT}_c$ approach finds a solution immediately across all satisfiable instances. As in

Downloaded from https://www.cambridge.org/core. UNIVERSIDAD AUTONOMA METROPOLITANA, on 08 Nov 2026 at 14:47:57, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://doi.org/10.1017/S1471068426100714

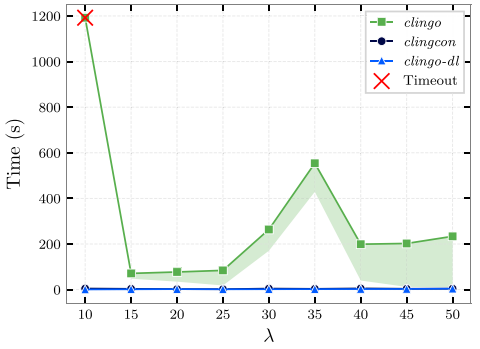



Fig. 10. Total times for *ft06* with relaxed time-point limit $\nu = 110$.

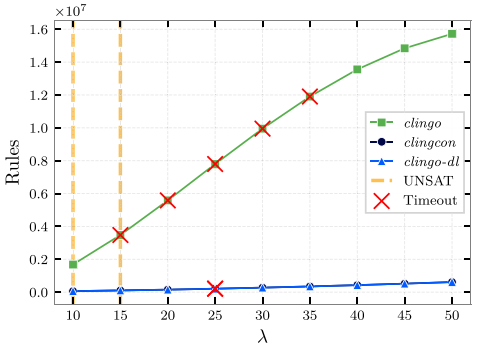



Fig. 11. Number of ground rules for *ft06* with time-point limit $\nu = 55$.

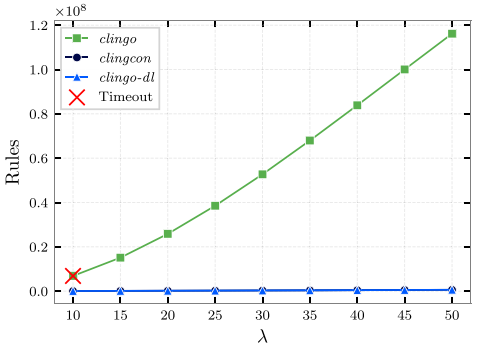



Fig. 12. Number of ground rules for *ft06* with relaxed time-point limit $\nu = 110$.

the previous applications, the grounding time for *clingo* grows with the horizon, so the relaxed time-point limit benefits smaller horizons where solutions can be found, but introduces overhead for larger ones. Figures 11 and 12 show that the number of ground rules for *clingo* grows linearly with $\lambda$, reflecting the propositional grounding of time-step constraints, whereas *clingcon* and *clingo*[DL] maintain a compact representation independent of the time-point limit.

Downloaded from https://www.cambridge.org/core. UNIVERSIDAD AUTONOMA METROPOLITANA, on 08 Nov 2026 at 14:47:57, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://doi.org/10.1017/S1471068426100714

## 6 Conclusion

We presented a computational approach to metric ASP that allows for fine-grained timing constraints. In doing so, we proposed two metric fragments. The first fragment restricts the use of metric operators in rule heads, allowing only the next operator. The expressiveness of this fragment showed to be enough to model basic transitions. However, it lacks the expressiveness needed for capturing global metric conditions. The second fragment, on the other hand, permits a broader use of global metric operators, significantly increasing expressiveness at the cost of higher computational complexity. For each fragment, we developed two alternative translations from firm semantic foundations, and proved their completeness and correctness. Our second translation has a clear edge over the first one, when it comes to a fine-grained resolution of time. This is achieved by outsourcing the treatment of time by using difference constraints. However, a prominent use case involves employing the identity (timing) function, where intervals reference only state indices within traces. This coarser notion of time reduces the discrepancy between our two translations. Further improvement is possible through more sophisticated Boolean encodings, such as an order encoding (Crawford and Baker 1994; Banbara *et al.* 2015).

We use meta-programming for implementation as it facilitates development and rapid prototyping for different fragments and translations. Our empirical analysis allowed us to compare translations and target systems, providing a basis for future work to improve the systems performance.

Overall our hybrid approach is superior to the Boolean one. Among the former, *clingo*[DL] outperforms *clingcon* in solving time across all benchmarks. Note however that the semantics developed in this work corresponds to that of *clingcon*, while *clingo*[DL] only computes canonical models (cf. Cabalar *et al.* (2023)). *Clingo*[DL] has furthermore the advantage that it does not require a fixed time-point limit in advance. Selecting such a limit is non-trivial in practice: its value is rarely known beforehand, it critically affects performance, and an inadequate choice may require multiple solver calls. Moreover, without a finite bound, *clingcon* cannot terminate its search since the timing function might be unbounded, while for *clingo* the time-point limit directly determines the size of the ground program, making its choice critical for grounding performance.

Outsourcing the treatment of time is not for free either. *Clingo*[DL] maps difference constraints into graphs, whose nodes are time variables and weighted edges reflect the actual constraints. This results in a quadratic space complexity. *Clingcon* pursues a lazy approach to constraint solving that gradually unfolds an ASP encoding of linear constraints. In the worst case, this amounts to the space requirements of our first translation. As well, such constraints are hidden from the ASP solver and cannot be used for directing the search. Hence, despite our indicative observations, a detailed empirical analysis is needed to account for the subtleties of our translation and its target systems. For instance, the irregular behavior of *clingcon* across instances and map layouts should be further investigated, as it suggests sensitivity to factors not yet fully understood.

## Competing interests

The authors declare none.

Downloaded from https://www.cambridge.org/core. UNIVERSIDAD AUTONOMA METROPOLITANA, on 08 Nov 2026 at 14:47:57, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://doi.org/10.1017/S1471068426100714

## References


Aguado, F., Cabalar, P., Diéguez, M., Pérez, G., Schaub, T., Schuhmann, A. and Vidal, C. 2023. Linear-time temporal answer set programming. *Theory and Practice of Logic Programming* 23, 1, 2–56. https://doi.org/10.1017/S1471068421000557.

Balduccini, M. 2009. Representing constraint satisfaction problems in answer set programming. In *Proceedings of the Second Workshop on Answer Set Programming and Other Computing Paradigms (ASPOCP'09)*, W. Faber and J. Lee, Eds. CEUR Workshop Proceedings, 16–30.

Balduccini, M., Magazzeni, D., Maratea, M. and Leblanc, E. 2017. CASP solutions for planning in hybrid domains. *Theory and Practice of Logic Programming* 17, 4, 591–633.

Banbara, M., Gebser, M., Inoue, K., Ostrowski, M., Peano, A., Schaub, T., Soh, T., Tamura, N. and Weise, M. 2015. aspartame: Solving constraint satisfaction problems with answer set programming. In *Proceedings of the Thirteenth International Conference on Logic Programming and Nonmonotonic Reasoning (LPNMR'15)*, F. Calimeri, G. Ianni and M. Truszczyński, Eds. 9345 of Lecture Notes in Artificial Intelligence, Springer-Verlag, 112–126.

Banbara, M., Kaufmann, B., Ostrowski, M. and Schaub, T. 2017. Clingcon: The next generation. *Theory and Practice of Logic Programming* 17, 4, 408–461. https://doi.org/10.1017/S1471068417000138.

Baselice, S., Bonatti, P. and Gelfond, M. 2005. Towards an integration of answer set and constraint solving. *Proceedings of the Twenty-first International Conference on Logic Programming (ICLP'05)*, M. Gabbrielli and G. Gupta, Eds. 3668 of Lecture Notes in Computer Science, Springer-Verlag, 52–66.

Beck, H., Dao-Tran, M. and Eiter, T. 2018. LARS: A logic-based framework for analytic reasoning over streams. *Artificial Intelligence* 261, 16–70. https://doi.org/10.1016/j.artint.2018.04.003.

Becker, A., Cabalar, P., Diéguez, M., Schaub, T. and Schuhmann, A. 2024. Metric temporal answer set programming over timed traces. *Theory and Practice of Logic Programming* 24, 3, 425–452. https://doi.org/10.1017/S1471068424000139.

Behrens, J., Kaminski, R., Schaub, T., Son, T., Švancara, J. and Wanko, P. 2024. Routing and scheduling in answer set programming applied to multi-agent path finding: Preliminary report. CoRR, abs/2403.12153. URL: https://arxiv.org/abs/2403.12153

Bellomarini, L., Blasi, L., Nissl, M. and Sallinger, E. 2022. The temporal vadalog system. In *Proceedings of the Sixth International Joint Conference on Rules and Reasoning (RuleML+RR'22)*, G. Governatori and A. Turhan, Eds. 13752 of Lecture Notes in Computer Science, Springer-Verlag, 130–145. https://doi.org/10.1007/978-3-031-21541-4

Ben-Eliyahu, R. 2021. How to split a logic program. In *Technical Communications of the Thirty-Seventh International Conference on Logic Programming (ICLP'21)*, A. Formisano, Y. Liu, B. Bogaerts, A. Brik, V. Dahl, C. Dodaro, P. Fodor, G. Pozzato, J. Vennekens and N. Zhou, Eds. 345 of Electronic Proceedings in Theoretical Computer Science (EPTCS), 27–40. https://doi.org/10.4204/EPTCS.345.13.

Cabalar, P., Pearce, D. and Valverde, A. 2005. Reducing propositional theories in equilibrium logic to logic programs. In *Proceedings of the Twelfth Portuguese Conference on Artificial Intelligence (EPIA'05)*, 3808 of Lecture Notes in Computer Science, Springer-Verlag, 4–17.

Cabalar, P., Kaminski, R., Ostrowski, M. and Schaub, T. 2016. An ASP semantics for default reasoning with constraints. In *Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence (IJCAI'16)*, S. Kambhampati, Ed. IJCAI/AAAI Press, 1015–1021. https://dl.acm.org/doi/10.5555/3060621.3060762.

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. https://doi.org/10.1017/S1471068418000297.


Downloaded from https://www.cambridge.org/core. UNIVERSIDAD AUTONOMA METROPOLITANA, on 08 Nov 2026 at 14:47:57, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://doi.org/10.1017/S1471068426100714

Cabalar, P., Diéguez, M., Laferriere, F. and Schaub, T. 2020a. Implementing dynamic answer set programming over finite traces. In *Proceedings of the Twenty-Fourth European Conference on Artificial Intelligence (ECAI'20)*, G. De Giacomo, A. Catalá, B. Dilkina, M. Milano, S. Barro, A. Bugarín and J. Lang, Eds. IOS Press, 656–663. https://doi.org/10.3233/FAIA200151.

Cabalar, P., Diéguez, M., Schaub, T. and Schuhmann, A. 2020b. Towards metric temporal answer set programming. *Theory and Practice of Logic Programming* 20, 5, 783–798.

Cabalar, P., Fandinno, J., Schaub, T. and Wanko, P. 2020c. A uniform treatment of aggregates and constraints in hybrid ASP. In *Proceedings of the Seventeenth International Conference on Principles of Knowledge Representation and Reasoning (KR'20)*, AAAI Press, 193–202. https://doi.org/10.24963/KR.2020/20.

Cabalar, P., Fandinno, J., Schaub, T. and Wanko, P. 2023. On the semantics of hybrid ASP systems based on clingo. *Algorithms* 16, 4, 185. https://doi.org/10.3390/a16040185

Crawford, J. and Baker, A. 1994. Experimental results on the application of satisfiability algorithms to scheduling problems. In *Proceedings of the Twelfth National Conference on Artificial Intelligence (AAAI'94)*, B. Hayes-Roth and R. Korf, Eds. AAAI Press, 1092–1097.

Cresswell, S. and Coddington, A. 2003. Planning with timed literals and deadlines. In *Proceedings of Twenty-second Workshop of the UK Planning and Scheduling Special Interest Group*, J. Porteous, Ed. University of Strathclyde, 23–35.

Fisher, M., Gabbay, D. and Vila, L. 2005. *Handbook of Temporal Reasoning in Artificial Intelligence*, 1 *of Foundations of Artificial Intelligence*. Elsevier Science.

Gebser, M., Kaminski, R., Kaufmann, B., Lindauer, M., Ostrowski, M., Romero, J., Schaub, T. and Thiele, S. 2015. *Potassco User Guide*, 2 ed. University of Potsdam. URL: http://potassco.org.

Gerevini, A., Haslum, P., Long, D., Saetti, A. and Dimopoulos, Y. 2009. Deterministic planning in the fifth international planning competition: PDDL3 and experimental evaluation of the planners. *Artificial Intelligence* 173, 5–6, 619–668.

Gödel, K. 1932. Zum intuitionistischen Aussagenkalkül. *Anzeiger der Akademie der Wissenschaften in Wien* 69, 65–66.

Hustadt, U., Ozaki, A. and Dixon, C. 2020. Theorem proving for pointwise metric temporal logic over the naturals via translations. *Journal of Automated Reasoning* 64, 8, 1553–1610. https://doi.org/10.1007/s10817-020-09541-4.

Janhunen, T., Kaminski, R., Ostrowski, M., Schellhorn, S., Wanko, P. and Schaub, T. 2017. Clingo goes linear constraints over reals and integers. *Theory and Practice of Logic Programming* 17, 5–6, 872–888. https://doi.org/10.1017/S1471068417000242.

Kaminski, R., Romero, J., Schaub, T. and Wanko, P. 2023. How to build your own ASP-based system?! *Theory and Practice of Logic Programming* 23, 1, 299–361. https://doi.org/10.1017/S1471068421000508.

Kamp, J. 1968. Tense Logic and the Theory of Linear Order. Ph.D. thesis, University of California at Los Angeles.

Lifschitz, V. 2019. *Answer Set Programming*. Springer-Verlag. https://doi.org/10.1007/978-3-030-24658-7

Lifschitz, V. and Turner, H. 1994. Splitting a logic program. In *Proceedings of the Eleventh International Conference on Logic Programming*, MIT Press, 23–37.

Mellarkod, V. 2007. Integrating ASP and CLP Systems: Computing Answer Sets from Partially Ground Programs. Ph.D. thesis, Texas Tech University.

Muth, J. and Thompson, G. 1963. Probabilistic learning combinations of local job-shop scheduling rules. Industrial scheduling.

Downloaded from https://www.cambridge.org/core. UNIVERSIDAD AUTONOMA METROPOLITANA, on 08 Nov 2026 at 14:47:57, subject to the Cambridge Core terms of use , available at https://www.cambridge.org/core/terms. https://doi.org/10.1017/S1471068426100714

Pearce, D. 1997. A new logical characterisation of stable models and answer sets. In *Proceedings of the Sixth International Workshop on Non-Monotonic Extensions of Logic Programming (NMELP'96)*, 1216 of Lecture Notes in Computer Science, Springer, 57–70. https://doi.org/10.1007/BFb0023801.

Son, T., Baral, C. and Tuan, L. 2004. Adding time and intervals to procedural and hierarchical control specifications. In *Proceedings of the Nineteenth National Conference on Artificial Intelligence (AAAI'04)*, D. McGuinness and G. Ferguson, Eds. AAAI Press, 92–97. URL: http://www.aaai.org/Library/AAAI/2004/aaai04-015.php

Stern, R., Sturtevant, N., Felner, A., Koenig, S., Ma, H., Walker, T., Li, J., Atzmon, D., Cohen, L., Kumar, T., Barták, R. and Boyarski, E. 2019. Multi-agent pathfinding: Definitions, variants, and benchmarks. In *Proceedings of the Twelfth International Symposium on Combinatorial Search (SOCS'19)*, P. Surynek and W. Yeoh, Eds. AAAI Press, 151–159.

Tseitin, G. 1968. On the complexity of derivation in the propositional calculus. *Zapiski nauchnykh seminarov LOMI* 8, 234–259.

Wałega, P., Grau, B. C., Kaminski, M. and Kostylev, E. 2019. DatalogMTL: Computational complexity and expressive power. In *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence (IJCAI'19)*, S. Kraus, Ed. ijcai.org, 1886–1892.

Wałega, P., Cucala, D. T., Kostylev, E. and Grau, B. C. 2021. DatalogMTL with negation under stable models semantics. In *Proceedings of the Eighteenth International Conference on Principles of Knowledge Representation and Reasoning (KR'21)*, M. Bienvenu, G. Lakemeyer and E. Erdem, Eds. AAAI Press, 609–618.

Wang, D., Hu, P., Wałega, P. and Grau, B. C. 2022. MeTeoR: Practical reasoning in datalog with metric temporal operators. In *Proceedings of the Thirty-sixth National Conference on Artificial Intelligence (AAAI'22)*, AAAI Press, 5906–5913. https://doi.org/10.1609/aaai.v36i5.20535.

## Appendix A. Three-valued semantics

In our framework, HT-interpretations (and -traces) naturally lend themselves to three-valued models, where each atom is classified as false, assumed, or proven (in a given state). This correspondence arises from the fact that the intermediate logic of HT aligns with Gödel's three-valued logic $G_3$ (Gödel 1932). Consequently, formula satisfaction in HT can be directly represented by a three-valued function, $\boldsymbol{m}(\alpha)$, assigning values 0 (false), 1 (assumed), or 2 (proven) to each formula $\alpha$. The significance of this multi-valued truth assignment lies in its ability to simplify HT-equivalence, reducing $\alpha \equiv \beta$ to a straightforward comparison of truth values, $\boldsymbol{m}(\alpha) = \boldsymbol{m}(\beta)$. This property proves invaluable when introducing auxiliary atoms, enabling the replacement of complex formulas with simpler atoms while ensuring semantic equivalence. Specifically, by introducing an auxiliary atom $a$ to represent $\alpha$, we can safely substitute $\alpha$ with $a$, provided we add formulas that guarantee $\boldsymbol{m}(\alpha) = \boldsymbol{m}(a)$. Building upon these principles, we now present the following formal definitions.

### *A.1 Three-valued semantics for HT*

The following definitions follow the ones in Cabalar *et al.* (2005).

*Definition 3 (Three-valued interpretation).*
A three-valued interpretation over $\mathcal{A}$ is a function $\boldsymbol{m} : \mathcal{A} \to \{0, 1, 2\}$.

A three-valued interpretation is total, if its range is $\{0, 2\}$.

Downloaded from https://www.cambridge.org/core. UNIVERSIDAD AUTONOMA METROPOLITANA, on 08 Nov 2026 at 14:47:57, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://doi.org/10.1017/S1471068426100714

*Definition 4 (Extension of three-valued interpretation).*
A three-valued interpretation $\boldsymbol{m}$ over $\mathcal{A}$ is extended to Boolean formulas as follows.

$$\begin{aligned}
\boldsymbol{m}(\bot) &= 0 \\
\boldsymbol{m}(\varphi \wedge \psi) &= \min(\boldsymbol{m}(\varphi), \boldsymbol{m}(\psi)) \\
\boldsymbol{m}(\varphi \vee \psi) &= \max(\boldsymbol{m}(\varphi), \boldsymbol{m}(\psi)) \\
\boldsymbol{m}(\varphi \to \psi) &= \begin{cases} 2 & \text{if } \boldsymbol{m}(\varphi) \leq \boldsymbol{m}(\psi) \\ \boldsymbol{m}(\psi) & \text{otherwise} \end{cases}
\end{aligned}$$

For the derived operators we have $\boldsymbol{m}(\top) = 2$ and $\boldsymbol{m}(\neg\varphi) = 2$ if $\boldsymbol{m}(\varphi) = 0$ and 0 otherwise.

*Proposition 7 (Satisfaction total three-valued interpretations).*
For any total three-valued interpretation $\boldsymbol{m}$ and for any propositional formula $\varphi$, $\boldsymbol{m}(\varphi) \in \{0, 2\}$.

*Proposition 8 (Equivalent formulas).*
For any propositional formulas $\varphi$ and $\psi$, $\varphi \equiv \psi$ iff $\boldsymbol{m}(\varphi) = \boldsymbol{m}(\psi)$ for any three-valued interpretation $\boldsymbol{m}$.

In order to prove that HT and its three-valued characterization are equivalent, we first define a mapping between HT and three-valued mappings. Then we prove that such correspondence can be extended to the full propositional language.

*Definition 5 (Correspondence between HT and three-valued interpretations).*
Given an HT interpretation $\langle H, T\rangle$ over $\mathcal{A}$, we define its corresponding three-valued interpretation $\boldsymbol{m}$ over $\mathcal{A}$ for each atom $a \in \mathcal{A}$ as

$$\boldsymbol{m}(a) = \begin{cases} 0 & \text{if } a \notin T \\ 1 & \text{if } a \in T \setminus H \\ 2 & \text{if } a \in H \end{cases}$$

Conversely, given a three-valued interpretation $\boldsymbol{m}$ over $\mathcal{A}$, we define its corresponding HT interpretation over $\mathcal{A}$ as $\langle H, T\rangle$, where

$$\begin{aligned}
T &= \{a \in \mathcal{A} \mid \boldsymbol{m}(a) > 0\} \\
H &= \{a \in \mathcal{A} \mid \boldsymbol{m}(a) = 2\}
\end{aligned}$$

*Proposition 9 (Equi-satisfaction).*
Given any HT interpretation $\langle H, T\rangle$ and any three-valued interpretation $\boldsymbol{m}$ satisfying the correspondence established in Definition 5, it follows that

$$\begin{aligned}
\boldsymbol{m}(\varphi) = 2 \;\text{ iff }\; \langle H, T\rangle &\models \varphi \text{ and} \\
\boldsymbol{m}(\varphi) > 0 \;\text{ iff }\; \langle T, T\rangle &\models \varphi
\end{aligned}$$

for all propositional formulas $\varphi$.

*Corollary 1 (Total three-valued correspondence).*
Let $\langle T, T\rangle$ a total HT interpretation and let $\boldsymbol{m}$ a three-valued interpretation satisfying

Downloaded from https://www.cambridge.org/core. UNIVERSIDAD AUTONOMA METROPOLITANA, on 08 Nov 2026 at 14:47:57, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://doi.org/10.1017/S1471068426100714

the correspondence established in Definition 5, it follows that

$$\boldsymbol{m}(\varphi) = 2 \text{ iff } \langle T, T\rangle \models \varphi \text{ and}$$
$$\boldsymbol{m}(\varphi) = 0 \text{ iff } \langle T, T\rangle \not\models \varphi.$$

for all propositional formulas $\varphi$.

### *A.2 Three-valued semantics for HT$_c$*

*Definition 6 (Corresponding three-valued interpretation).*
Given an HT$_c$ interpretation $\langle h, t\rangle$ and a denotation $[\![\cdot]\!]$ over $\langle \mathcal{X}, \mathcal{D}, \mathcal{C}\rangle$, we define its corresponding three-valued interpretation $\boldsymbol{m}^{\langle h,t\rangle}$ over $\mathcal{C}$ for each constraint atom $c \in \mathcal{C}$ as[11]

$$\boldsymbol{m}^{\langle h,t\rangle}(c) = \begin{cases} 2 & \text{if } h \in [\![ c ]\!] \\ 1 & \text{if } h \notin [\![ c ]\!] \text{ and } t \in [\![ c ]\!] \\ 0 & \text{if } t \notin [\![ c ]\!] \end{cases}$$

Note that $\boldsymbol{m}^{\langle h,t\rangle}$ is total when $h = t$.

*Definition 7 (Extension of three-valued interpretation).*
A three-valued interpretation $\boldsymbol{m}^{\langle h,t\rangle}$ corresponding to an HT$_c$ interpretation $\langle h, t\rangle$, is extended to Boolean constraint formulas as follows.

$$\begin{aligned} \boldsymbol{m}^{\langle h,t\rangle}(\bot) &= 0 \\ \boldsymbol{m}^{\langle h,t\rangle}(\varphi \wedge \psi) &= \min\{\boldsymbol{m}^{\langle h,t\rangle}(\varphi), \boldsymbol{m}^{\langle h,t\rangle}(\psi)\} \\ \boldsymbol{m}^{\langle h,t\rangle}(\varphi \vee \psi) &= \max\{\boldsymbol{m}^{\langle h,t\rangle}(\varphi), \boldsymbol{m}^{\langle h,t\rangle}(\psi)\} \\ \boldsymbol{m}^{\langle h,t\rangle}(\varphi \to \psi) &= \begin{cases} 2 & \text{if } \boldsymbol{m}^{\langle h,t\rangle}(\varphi) \leq \boldsymbol{m}^{\langle h,t\rangle}(\psi) \\ \boldsymbol{m}^{\langle h,t\rangle}(\psi) & \text{otherwise} \end{cases} \end{aligned}$$

For the derived operators we have $\boldsymbol{m}^{\langle h,t\rangle}(\top) = 2$ and $\boldsymbol{m}^{\langle h,t\rangle}(\neg\varphi) = 2$ if $\boldsymbol{m}^{\langle h,t\rangle}(\varphi) = 0$ and 0 otherwise.

*Proposition 10 (Satisfaction total three-valued interpretations).*
For any total three-valued interpretation $\boldsymbol{m}^{\langle t,t\rangle}$ and for any formula $\varphi$, $\boldsymbol{m}^{\langle t,t\rangle}(\varphi) \in \{0, 2\}$.

*Proof.*
Proven by straightforward structural induction on $\varphi$. □

*Proposition 11 (Equivalence of formulas).*
For any propositional formulas $\varphi$ and $\psi$, $\varphi \equiv \psi$ iff $\boldsymbol{m}^{\langle h,t\rangle}(\varphi) = \boldsymbol{m}^{\langle h,t\rangle}(\psi)$ for any $\langle h, t\rangle$.

*Proof.*
Suppose $\varphi \equiv \psi$ but there is some $\boldsymbol{m}^{\langle h,t\rangle}$ such that $\boldsymbol{m}^{\langle h,t\rangle}(\varphi) \neq \boldsymbol{m}^{\langle h,t\rangle}(\psi)$. By the three-valued semantics, $\boldsymbol{m}^{\langle h,t\rangle}(\varphi \leftrightarrow \psi) \neq 2$, This contradicts the fact that $\varphi \equiv \psi$. The converse direction is proven analogously. □

[11] Since we use strict denotations, it is sufficient to check in the first case only satisfaction in $h$.

Downloaded from https://www.cambridge.org/core. UNIVERSIDAD AUTONOMA METROPOLITANA, on 08 Nov 2026 at 14:47:57, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://doi.org/10.1017/S1471068426100714

To prove that $\mathrm{HT}_c$ and our (alternative) three valued characterization are equivalent, there is no need to establish a model correspondence since the three-valued characterization is induced by and $\mathrm{HT}_c$ interpretation. Consequently, the following lemma for the full propositional language can be directly proved.

*Proposition 12 (Equi-satisfaction).*
Given an $\mathrm{HT}_c$ interpretation $\langle h, t\rangle$ , it follows that

$$\begin{aligned} \boldsymbol{m}^{\langle h,t\rangle}(\varphi) = 2 \;\; &\text{iff} \;\; \langle h, t\rangle \models \varphi \text{ and} \\ \boldsymbol{m}^{\langle h,t\rangle}(\varphi) > 0 \;\; &\text{iff} \;\; \langle t, t\rangle \models \varphi. \end{aligned}$$

for all propositional formulas $\varphi$.

*Proof.*
Proven by structural induction on $\varphi$, taking into account that we have two induction hypotheses, one for each of the two cases in the proposition. □

*Proposition 13 (Total three-valued correspondence).*
Given a total $\mathrm{HT}_c$ interpretation $\langle t, t\rangle$, it follows that

$$\begin{aligned} \boldsymbol{m}^{\langle t,t\rangle}(\varphi) = 2 \;\; &\text{iff} \;\; \langle t, t\rangle \models \varphi \text{ and} \\ \boldsymbol{m}^{\langle t,t\rangle}(\varphi) = 0 \;\; &\text{iff} \;\; \langle t, t\rangle \not\models \varphi. \end{aligned}$$

for all propositional formulas $\varphi$.

*Proof.*
Follows from Proposition 12. □

### *A.3 Three-valued semantics for MHT*

*Definition 8 (Three-valued metric interpretation).*
Given a timing function $\tau : [0..\lambda) \to \mathbb{N}$ with $\lambda \in \mathbb{N}$, we define a three-valued metric interpretation $\boldsymbol{m}^\tau$ over alphabet $\mathcal{A}$ for as the function

$$\boldsymbol{m}^\tau : [0..\lambda) \times \mathcal{A} \to \{0, 1, 2\}$$

The three-valued interpretation $\boldsymbol{m}^\tau$ assigns to each state $k \in [0..\lambda)$ and each atom $a \in \mathcal{A}$ a truth value in $\{0, 1, 2\}$. In analogy to Section 2.3, the associated time function $\tau$ captures the time of each state.

A three-valued metric interpretation $\boldsymbol{m}^\tau$ is total, if its range is $\{0, 2\}$ for all $k \in [0..\lambda)$.

*Definition 9 (Extension of three-valued metric interpretation).*
A three-valued metric interpretation $\boldsymbol{m}^\tau$, is extended to metric formulas for any time point $k \in [0..\lambda)$ as follows.

$$\begin{aligned} \boldsymbol{m}^\tau(k, \bot) &= 0 \\ \boldsymbol{m}^\tau(k, \varphi \wedge \psi) &= \min\{\boldsymbol{m}^\tau(k, \varphi), \boldsymbol{m}^\tau(k, \psi)\} \\ \boldsymbol{m}^\tau(k, \varphi \vee \psi) &= \max\{\boldsymbol{m}^\tau(k, \varphi), \boldsymbol{m}^\tau(k, \psi)\} \\ \boldsymbol{m}^\tau(k, \varphi \to \psi) &= \begin{cases} 2 & \text{if } \boldsymbol{m}^\tau(k, \varphi) \leq \boldsymbol{m}^\tau(k, \psi) \\ \boldsymbol{m}^\tau(k, \psi) & \text{otherwise} \end{cases} \end{aligned}$$

Downloaded from https://www.cambridge.org/core. UNIVERSIDAD AUTONOMA METROPOLITANA, on 08 Nov 2026 at 14:47:57, subject to the Cambridge Core terms of use , available at https://www.cambridge.org/core/terms. https://doi.org/10.1017/S1471068426100714

$$\boldsymbol{m}^{\tau}(k, \mathbf{I}) = \begin{cases} 2 & \text{if } k = 0 \\ 0 & \text{otherwise} \end{cases}$$

$$\boldsymbol{m}^{\tau}(k, \bigcirc_I \varphi) = \begin{cases} 0 & \text{if } k+1 = \lambda \text{ or } \tau(k+1) - \tau(k) \notin I \\ \boldsymbol{m}^{\tau}(k+1, \varphi) & \text{otherwise} \end{cases}$$

$$\boldsymbol{m}^{\tau}(k, \Diamond_I \varphi) = \max\{0\} \cup \{\boldsymbol{m}^{\tau}(j, \varphi) \mid k \leq j < \lambda \text{ and } \tau(j) - \tau(k) \in I\}$$

$$\boldsymbol{m}^{\tau}(k, \Box_I \varphi) = \min\{2\} \cup \{\boldsymbol{m}^{\tau}(j, \varphi) \mid k \leq j < \lambda \text{ and } \tau(j) - \tau(k) \in I\}$$

Unlike in the three-valued semantics of THT$_f$ (Cabalar *et al.* 2018), the sets for eventually and always might be empty due to the restrictions on the interval. Therefore, we add the default values, 0 and 2, respectively.

*Proposition 14 (Satisfaction total three-valued metric interpretations).*
For any total three-valued valuation $\boldsymbol{m}^{\tau}$, for any metric formula $\varphi$ and for all $k \in [0..\lambda)$, $\boldsymbol{m}^{\tau}(k, \varphi) \in \{0, 2\}$.

*Proof.*
Proven by straightforward structural induction on $\varphi$. □

*Proposition 15 (Point-wise equivalence for formulas).*
Given a timed HT-trace $\mathbf{M} = (\langle \mathbf{H}, \mathbf{T} \rangle, \tau)$ and its corresponding three-valued metric interpretation $\boldsymbol{m}^{\tau}$, for any arbitrary metric formulas $\varphi$ and $\psi$, and $k \in [0..\lambda)$, it follows that $\mathbf{M}, k \models \varphi \leftrightarrow \psi$ iff $\boldsymbol{m}^{\tau}(k, \varphi) = \boldsymbol{m}^{\tau}(k, \psi)$.

*Proof.*
$\mathbf{M}, k \models \varphi \leftrightarrow \psi$ iff $\boldsymbol{m}^{\tau}(\varphi \leftrightarrow \psi) = 2$. We know this is the case iff $\boldsymbol{m}^{\tau}(k, \varphi) = \boldsymbol{m}^{\tau}(k, \psi)$ by the definition of the metric three-valued semantics. □

*Proposition 16 (Equivalent formulas).*
Two arbitrary metric formulas $\varphi$ and $\psi$ are equivalent, in symbols $\varphi \equiv \psi$, iff $\boldsymbol{m}^{\tau}(k, \varphi) = \boldsymbol{m}^{\tau}(k, \psi)$ for any three-valued metric interpretation $\boldsymbol{m}^{\tau}$, any $k \in [0..\lambda)$ and any timed function $\tau$.

*Proof.*
Suppose $\varphi \equiv \psi$ but there is some timed function $\tau$ and $\boldsymbol{m}^{\tau}$ such that for some $k \in [0..\lambda)$, $\boldsymbol{m}^{\tau}(k, \varphi) \neq \boldsymbol{m}^{\tau}(k, \psi)$. By the three-valued semantics, $\boldsymbol{m}^{\tau}(k, \varphi \leftrightarrow \psi) \neq 2$, This contradicts the fact that $\varphi \equiv \psi$. The converse direction is proven analogously. □

As in the propositional case, we prove that MHT and our three-valued characterization are equivalent. We start by defining a model correspondence.

*Definition 10 (Correspondence between MHT and three-valued metric interpretations).*
Given a timed HT-trace $\mathbf{M} = (\langle \mathbf{H}, \mathbf{T} \rangle, \tau)$ of length $\lambda$ over $\mathcal{A}$, we define its corresponding three-valued metric interpretation $\boldsymbol{m}^{\tau}$ over $\mathcal{A}$ as

$$\boldsymbol{m}^{\tau}(k, a) = \begin{cases} 0 & \text{if } a \notin T_k \\ 1 & \text{if } a \in T_k \setminus H_k \\ 2 & \text{if } a \in H_k \end{cases}$$

Downloaded from https://www.cambridge.org/core. UNIVERSIDAD AUTONOMA METROPOLITANA, on 08 Nov 2026 at 14:47:57, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://doi.org/10.1017/S1471068426100714

for $a \in \mathcal{A}$ and $k \in [0..\lambda)$. Conversely, given a three-valued metric interpretation $\boldsymbol{m}^\tau$ over $\mathcal{A}$, we define its associated timed HT-trace over $\mathcal{A}$ as $\mathbf{M} = (\langle \mathbf{H}, \mathbf{T} \rangle, \tau)$ of length $\lambda$ as

$$H_k = \{a \in \mathcal{A} \mid \boldsymbol{m}^\tau(k, a) = 2\}$$
$$T_k = \{a \in \mathcal{A} \mid \boldsymbol{m}^\tau(k, a) > 0\}$$

for every $k \in [0..\lambda)$.

*Proposition 17 (Equi-satisfaction).*
Given a timed HT-trace $\mathbf{M} = (\langle \mathbf{H}, \mathbf{T} \rangle, \tau)$ and a three-valued metric interpretation $\boldsymbol{m}^\tau$ satisfying the correspondences established in Definition 10, it follows that

$$\boldsymbol{m}^\tau(k, \varphi) = 2 \text{ iff } \mathbf{M}, k \models \varphi$$
$$\boldsymbol{m}^\tau(k, \varphi) > 0 \text{ iff } (\langle \mathbf{T}, \mathbf{T} \rangle, \tau), k \models \varphi.$$

for all metric formulas $\varphi$ and any $k \in [0..\lambda)$.

*Proof.*
Proven by structural induction on $\varphi$, taking into account that we have two induction hypotheses, one for each of the two cases in the proposition. □

*Corollary 2 (Total three-valued metric correspondence).*
Let $(\langle \mathbf{H}, \mathbf{T} \rangle, \tau)$ a total timed HT-trace and let $\boldsymbol{m}^\tau$ a three-valued metric interpretation satisfying the correspondences established in Definition 10, it follows that

$$\boldsymbol{m}^\tau(k, \varphi) = 2 \;\text{ iff }\; (\langle \mathbf{H}, \mathbf{T} \rangle, \tau) \models \varphi \text{ and}$$
$$\boldsymbol{m}^\tau(k, \varphi) = 0 \;\text{ iff }\; (\langle \mathbf{H}, \mathbf{T} \rangle, \tau) \not\models \varphi.$$

for all metric formulas $\varphi$ and any $k \in [0..\lambda)$.

*Proof.*
Follows from Proposition 17. □

## Appendix B. Splitting

*Definition 11 (Splitting set Lifschitz and Turner (1994); Ben-Eliyahu (2021)).*
Let $P$ be a logic program and let $U$ be a set of atoms. $U$ is said to be a splitting set for $P$ if for all rule $r \in P$, if one of the atoms in the head of $r$ is in U then all the atoms of $r$ are in $U$. We denote by $b_U(P)$ the set of rules in $P$ having only atoms from $U$.

*Definition 12 (Reducing a logic program Ben-Eliyahu (2021)).*
Let $P$ be a logic program and let $X, Y$ be two sets of atoms. the procedure $Reduce(P, X, Y)$ returns a logic program obtained from $P$ in which all atoms in $X$ are set to $true$ and the atoms in $Y$ are set to $false$. The procedure $Reduce(P, X, Y)$ is shown in Algorithm 1.

*Theorem 10 (Adopted from Lifschitz and Turner (1994); Ben-Eliyahu (2021)).*
Let $P$ be a logic program, and let $U$ be a splitting set for $P$. A set of atoms $S$ is a stable model for $P$ if and only if $X = X \cup Y$ where $X$ is a stable model of $b_U(P)$ and $Y$ is a stable model of $Reduce(P \setminus b_U, X, U \setminus X)$.

Downloaded from https://www.cambridge.org/core. UNIVERSIDAD AUTONOMA METROPOLITANA, on 08 Nov 2026 at 14:47:57, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://doi.org/10.1017/S1471068426100714

**Algorithm 1.** Reduce $(P,\ X,\ Y)$

**input** : A program $P$ and two sets of atoms: $X$ and $Y$
**output** : An update of $P$ assuming that all the atoms in $X$ are true and all atoms in $Y$ are false.

**foreach** *atom* $a \in X$ **do**
  **foreach** *rule* $r \in P$ **do**
    **if** *a appears negative in the body of* $r$ **then** delete $r$;
    **else if** *a appears in the head of* $r$ **then** delete $r$;
    **else** delete each positive appearance of $a$ in the body of $r$;
**foreach** *atom* $a \in Y$ **do**
  **foreach** *rule* $r \in P$ **do**
    **if** *a appears positive in the body of* $r$ **then** delete $r$;
    **else if** *a appears in the head of* $r$ **then** delete a from the head of $r$;
    **else** delete each negative appearance of $a$ in the body of $r$;

## Appendix C. Proofs

### C.1 Plain metric logic programs

*Definition 13 (Correspondence HT to MHT Trace).*
Given an HT interpretation $\langle H, T\rangle$ over $\mathcal{A}^*$ we define $\sigma_{\mathcal{A}}(\langle H, T\rangle)$ as the HT-trace

$$\langle \{a \in \mathcal{A} \mid a_k \in H\}, \{a \in \mathcal{A} \mid a_k \in T\}\rangle_{k\in[0..\lambda)}$$

*Proposition 18 (Next relaxed).*
Given a metric formula $\varphi$, and a timed trace $\mathbf{M}$, if $\mathbf{M}, k \models \bigcirc_I \varphi$, then $\mathbf{M}, k \models \bigcirc\varphi$.[12]

*Proof.*
If $\mathbf{M}, k \models \bigcirc_I \varphi$, then $\mathbf{M}, k{+}1 \models \varphi$ and and $k+1 < \lambda$, using MHT semantics. Given that with the definition of $\in [0..\omega)$ we have $\tau(k{+}1) - \tau(k) \in [0..\omega)$, then $\mathbf{M} \models \bigcirc\varphi$. □

*Definition 14 (Relaxed intervals for metric formulas).*
Given a metric formulas $\varphi$, we define $\Omega(\varphi)$ as the metric logic program obtained by replacing all intervals in $\varphi$ with $[0..\omega)$.

*Definition 15 (Relaxed intervals for Metric Logic Programs).*
Given a metric logic program $P$, we define $\Omega(P)$ as the metric logic program obtained applying $\Omega(r)$ to every rule $r$ in $P$.

*Lemma 1 (Metric formulas with relaxed intervals MHT to HT).*
Let $\varphi$ be a metric formula whose only temporal operator is metric next of form $\bigcirc_{[0..\omega)}$, and let $\mathbf{M}$ be a timed HT-trace of length $\lambda \in \mathbb{N}$. Then, for any $k \in [0..\lambda)$, if $\mathbf{M}, k \models \varphi$ then $\theta(\mathbf{M}) \models (\varphi)_k$.

[12] Notice the absence of an interval in MHT is understood as abbreviation for the fixed interval $[0..\omega)$ according to Becker et al. (2024)

Downloaded from https://www.cambridge.org/core. UNIVERSIDAD AUTONOMA METROPOLITANA, on 08 Nov 2026 at 14:47:57, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://doi.org/10.1017/S1471068426100714

*Proof.*
By structural induction on $\varphi$. Let $\mathbf{M} = (\langle H_k, T_k\rangle_{k\in[0..\lambda)}, \tau)$ and $\theta(\mathbf{M}) = \langle H, T\rangle$

$\boldsymbol{\varphi = \top}$: Since $\langle H, T\rangle \models \top$ by definition, the result follows.

$\boldsymbol{\varphi = \bot}$: This leads to a contradiction, as $\mathbf{M}, k \not\models \bot$.

$\boldsymbol{\varphi = \mathsf{I}}$: If $k = 0$, then $\langle H, T\rangle \models \top$ by the HT semantics, which implies that $\langle H, T\rangle \models (\mathsf{I})_k$. If $k > 0$, this leads to a contradiction, as $\mathbf{M}, k \not\models \mathsf{I}$.

$\boldsymbol{\varphi = \mathsf{F}}$: The reasoning is analogous to the case $\varphi = \mathsf{I}$.

$\boldsymbol{\varphi = a \in \mathcal{A}}$: By the MHT semantics, $a$ must be in $H_k$. By definition of $\theta()$, this means that $a_k \in H$. By the HT semantics, we conclude that $\langle H, T\rangle \models a_k$, which implies that $\langle H, T\rangle \models (a)_k$.

$\boldsymbol{\varphi = \bigcirc_{[0..\omega)} a}$: Since $\mathbf{M}, k \models \bigcirc_{[0..\omega)} a$, we analyze two cases: If $k = \lambda - 1$, we get a contradiction due to the MHT semantics. If $k < \lambda - 1$, then $\mathbf{M}, k+1 \models a$ and $\tau(k+1) - \tau(k) \in [0..\omega)$. By induction, this implies that $\langle H, T\rangle \models (a)_{k+1}$, which means that $\langle H, T\rangle \models (\bigcirc_{[0..\omega)} a)_k$.

$\boldsymbol{\varphi = \neg\psi}$: Since $\mathbf{M}, k \not\models \psi$, we get that $\langle H, T\rangle \not\models (\psi)_k$ by induction. By the HT semantics, this implies that $\langle H, T\rangle \models \neg(\psi)_k$, which means that $\langle H, T\rangle \models (\neg\psi)_k$.

$\boldsymbol{\varphi = \psi_1 \wedge \psi_2}$: Since $\mathbf{M}, k$ satisfies both $\psi_1$ and $\psi_2$, we get by induction that $\langle H, T\rangle$ satisfies both $(\psi_1)_k$ and $(\psi_2)_k$. By the HT semantics, this means that $\langle H, T\rangle \models (\psi_1)_k \wedge (\psi_2)_k$, which implies that $\langle H, T\rangle \models (\psi_1 \wedge \psi_2)_k$.

$\boldsymbol{\varphi = \psi_1 \vee \psi_2}$: The reasoning is analogous to the case $\varphi = \psi_1 \wedge \psi_2$.

$\boldsymbol{\varphi = \psi \leftarrow \beta}$: If $\mathbf{M}, k \not\models \beta$, then by induction, $\langle H, T\rangle \not\models (\beta)_k$. By the HT semantics, this implies that $\langle H, T\rangle \models (\psi)_k \leftarrow (\beta)_k$, which means that $\langle H, T\rangle \models (\psi \leftarrow \beta)_k$.
If $\mathbf{M}, k \models \beta$, then by induction, $\langle H, T\rangle \models (\beta)_k$. Since $\mathbf{M}, k \models \psi$, we conclude that $(\langle T_k, T_k\rangle_{k\in[0..\lambda)}, \tau), k \models \psi$. By induction, this means that both $\langle H, T\rangle$ and $\langle T, T\rangle$ satisfy $(\psi)_k$. By the HT semantics, we conclude that $\langle H, T\rangle \models (\psi)_k \leftarrow (\beta)_k$, which means that $\langle H, T\rangle \models (\psi \leftarrow \beta)_k$. □

*Lemma 2 (Metric formulas with relaxed intervals HT to MHT).*
Let $\varphi$ be a metric formula whose only temporal operator is metric next of form $\bigcirc_{[0..\omega)}$, and let $\langle H, T\rangle$ a HT interpretation such that for any $k \in [0..\lambda)$ $\langle H, T\rangle \models (\varphi)_k$, then, $(\sigma_{\mathcal{A}}(\langle H, T\rangle), \tau) \models \varphi$ for any strict timing function $\tau$.

*Proof.*
By structural induction on $\varphi$. Let $\mathbf{M} = (\sigma_{\mathcal{A}}(\langle H, T\rangle), \tau) = (\langle H_k, T_k\rangle_{k\in[0..\lambda)}, \tau)$.

$\boldsymbol{\varphi = \top}$: Since $\mathbf{M}, k \models \top$ by definition, the result follows.

$\boldsymbol{\varphi = \bot}$: This leads to a contradiction, as $\langle H, T\rangle \not\models \bot$.

$\boldsymbol{\varphi = \mathsf{I}}$: If $k = 0$, then $\mathbf{M}, k \models \mathsf{I}$ by the MHT semantics. If $k > 0$, then $\langle H, T\rangle \models (\mathsf{I})_k$, but this contradicts the fact that $\langle H, T\rangle \not\models \bot$.

Downloaded from https://www.cambridge.org/core. UNIVERSIDAD AUTONOMA METROPOLITANA, on 08 Nov 2026 at 14:47:57, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://doi.org/10.1017/S1471068426100714

$\boldsymbol{\varphi = \mathbf{F}}$: The reasoning is analogous to the case $\varphi = \mathbf{I}$.

$\boldsymbol{\varphi = a \in \mathcal{A}}$: Since $\langle H, T\rangle \models a_k$ by definition of $(r)_k$, we get that $a_k \in H$ by the HT semantics. By the definition of $\sigma_{\mathcal{A}}()$, this implies that $a \in H_k$. Therefore, by the MHT semantics, we conclude that $\mathbf{M}, k \models a$.

$\boldsymbol{\varphi = \bigcirc_{[0..\omega)} a}$: Since $\langle H, T\rangle \models (\bigcirc_{[0..\omega)} a)_k$, we analyze two cases: If $k = \lambda - 1$, then $\langle H, T\rangle \models \bot$, which is a contradiction. If $k < \lambda - 1$, then by the definition of $(r)_k$, we get that $\langle H, T\rangle \models (a)_{k+1}$. By induction, this implies that $\mathbf{M}, k+1 \models a$. Additionally, since $\tau(k+1) - \tau(k) \in [0..\omega)$ (which holds for any strict timing function), we conclude by the MHT semantics that $\mathbf{M}, k \models \bigcirc_{[0..\omega)} a$.

$\boldsymbol{\varphi = \neg\psi}$: Since $\langle H, T\rangle \models (\neg\psi)_k$, we know that $\langle H, T\rangle \models \neg(\psi)_k$ by the definition of $(r)_k$. This means that $\langle H, T\rangle \not\models (\psi)_k$ by the HT semantics. By induction, this implies that $\mathbf{M}, k \not\models \psi$. Therefore, by the MHT semantics, we conclude that $\mathbf{M}, k \models \neg\psi$.

$\boldsymbol{\varphi = \psi_1 \wedge \psi_2}$: Since $\langle H, T\rangle \models (\psi_1 \wedge \psi_2)_k$, we get that $\langle H, T\rangle \models (\psi_1)_k \wedge (\psi_2)_k$ by the definition of $(r)_k$. This implies that $\langle H, T\rangle \models (\psi_1)_k$ and $\langle H, T\rangle \models (\psi_2)_k$ by the HT semantics. By induction, we conclude that $\mathbf{M}, k \models \psi_1$ and $\mathbf{M}, k \models \psi_2$. Therefore, by the MHT semantics, we get that $\mathbf{M}, k \models \psi_1 \wedge \psi_2$.

$\boldsymbol{\varphi = \psi_1 \vee \psi_2}$: The reasoning is analogous to the case $\varphi = \psi_1 \wedge \psi_2$.

$\boldsymbol{\varphi = \psi \leftarrow \beta}$: Since $\langle H, T\rangle \models (\psi \leftarrow \beta)_k$, we get that $\langle H, T\rangle \models (\psi)_k \leftarrow (\beta)_k$ by the definition of $(r)_k$.

If $\langle H, T\rangle \not\models (\beta)_k$, then by induction, $\mathbf{M}, k \not\models \beta$. By the MHT semantics, this means that $\mathbf{M}, k \models \psi \leftarrow \beta$.

If $\langle H, T\rangle \models (\beta)_k$, then by induction, $\mathbf{M}, k \models \beta$. Since $\langle H, T\rangle \models (\psi)_k$ and $\langle T, T\rangle \models (\psi)_k$ by the HT semantics, we conclude by induction that $\mathbf{M}, k \models \psi$ and $(\langle T_k, T_k\rangle_{k\in[0..\lambda)}, \tau), k \models \psi$. Therefore, by the MHT semantics, we conclude that $\mathbf{M}, k \models \psi \leftarrow \beta$. □

*Theorem 11 (Temporal completeness with relaxed intervals).*
Let $P$ be a metric logic program and $\mathbf{M} = (\langle \mathbf{H}, \mathbf{T}\rangle, \tau)$ a timed HT-trace of length $\lambda$.
If $\mathbf{M} \models \Omega(P)$, then $\theta(\mathbf{M}) \models \Pi_\lambda(P)$

*Proof.*
We must prove that $\theta(\mathbf{M}) \models \bigcup_{\Box r \in P,\, 0 \le k < \lambda} (r)_k$ Since $\mathbf{M} \models \Omega(P)$, we have $\mathbf{M} \models \Omega(\Box r)$ for all $\Box r \in P$. MHT Semantics then gives us $\mathbf{M}, k \models \Omega(r)$ for all $\Box r \in P, 0 \le k < \lambda$, and by applying Lemma 1, we get $\theta(\mathbf{M}) \models (\Omega(r))_k$ for all $\Box r \in P, 0 \le k < \lambda$. Since $(\Omega(r))_k = (r)_k$, because intervals are irrelevant in $(r)_k$, we conclude that $\theta(\mathbf{M}) \models (r)_k$ for all $\Box r \in P, 0 \le k < \lambda$, which is the definition of $\Pi_\lambda(P)$. □

*Corollary 3 (Completeness without relaxed intervals).*
Let $P$ be a metric logic program and $\mathbf{M} = (\langle \mathbf{H}, \mathbf{T}\rangle, \tau)$ a timed HT-trace of length $\lambda$.
If $\mathbf{M} \models P$, then $\theta(\mathbf{M}) \models \Pi_\lambda(P)$

*Proof.*
We notice that the only difference form $\Omega(P)$ to $P$ is the $\bigcirc_I$ metric operator. As stated in Proposition 18, if $\mathbf{M}, k \models \bigcirc_I \varphi$, then $\mathbf{M}, k \models \bigcirc \varphi$. Therefore, know that if $\mathbf{M} \models P$ then $\mathbf{M} \models \Omega(P)$, so, we can apply Theorem 11. □

Downloaded from https://www.cambridge.org/core. UNIVERSIDAD AUTONOMA METROPOLITANA, on 08 Nov 2026 at 14:47:57, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://doi.org/10.1017/S1471068426100714

*Theorem 12 (Temporal correctness with relaxed intervals).*
Let $P$ be a metric logic program over $\mathcal{A}$ and let $\langle H, T\rangle$ be an HT interpretation over $\mathcal{A}^*$.

If $\langle H, T\rangle \models \Pi_\lambda(P)$, then $(\sigma_\mathcal{A}(\langle H, T\rangle), \tau) \models \Omega(P)$ for any strict timing function $\tau$.

*Proof.*
We take a strict timing function $\tau$ and let $\mathbf{M} = (\sigma_\mathcal{A}(\langle H, T\rangle), \tau)$. We must prove that $\mathbf{M} \models \Omega(P)$. We know $\langle H, T\rangle \models \Pi_\lambda(P)$, so $\langle H, T\rangle \models (r)_k$ for all $\Box r \in P, 0 \leq k < \lambda$. Since intervals are irrelevant in $(r)_k$, then $\langle H, T\rangle \models (\Omega(r))_k$ for all $\Box r \in P, 0 \leq k < \lambda$. By applying Lemma 2, we get $\mathbf{M}, k \models \Omega(r)$ for all $\Box r \in P, 0 \leq k < \lambda$, which using MHT Semantics, gives us $\mathbf{M} \models \Omega(P)$ □

### *C.1.1 Translation of plain metric logic programs to HT*

*Lemma 3 (Splitting Plain HT).*
Given a metric logic program $P$ and a $\lambda, \nu \in \mathbb{N}$, the stable models of $\Pi_\lambda(P) \cup \Delta_{\lambda,\nu} \cup \Psi_{\lambda,\nu}(P)$ can be computed by means of the splitting technique.

*Proof.*
We will show how the stable models of $\mathbb{P} = \Pi_\lambda(P) \cup \Delta_{\lambda,\nu} \cup \Psi_{\lambda,\nu}(P)$ can be computed by sequentially applying the splitting theorem. Let us start by fixing $U_0 = \{t_{k,d} \mid k, d \in \mathbb{N}\}$. Clearly it is an splitting set because the elements of $U_0$ are only generated by $\Delta_{\lambda,\nu}$. Moreover, $b_{U_0} = \Delta_{\lambda,\nu}$. By Theorem 10, an stable model $T$ of $\mathbb{P}$ can be computed as, $T = T_0 \cup T_1$ where $T_0$ is a stable model of $b_{U_0}$. $T_1$ is a stable model of $Reduce(\mathbb{P} \setminus b_{U_0}, T_0, U_0 \setminus T_0)$. If we analyze this first simplification, we will see that $Reduce(\mathbb{P} \setminus b_{U_0}, T_0, U_0 \setminus T_0) = \Pi_\lambda(P) \cup Reduce(\Psi_{\lambda,\nu}(P), T_0, U_0 \setminus T_0)$, because $\Pi_\lambda(P)$ does not contain atoms in $U_0$. More precisely, $Reduce(\Psi_{\lambda,\nu}(P), T_0, U_0 \setminus T_0)$ are just constraints that use only atoms from $\Pi_\lambda(P)$. □

*Proposition 19 (Model if timed equilibrium).*
Let $\langle H, T\rangle$ be an HT interpretation and $\lambda \in \mathbb{N}$.

If $\langle H, T\rangle$ is timed wrt $\lambda$ and induces $\tau$ then $\langle H, T\rangle|_\mathcal{T}$ is a total HT interpretation and an equilibrium model of $\Delta_{\lambda,\nu}$ for $\nu = \tau(\lambda - 1)$.

*Proof.*
The satisfaction follows from Proposition 2. From the definition of being timed it follows that $\langle H, T\rangle|_\mathcal{T}$ must be total. Now we prove that it is in equilibrium. Assume towards a contradiction that there is $\langle H, T\rangle$ such that $H \subset T$ and $\langle H, T\rangle \models \Delta_{\lambda,\nu}$. Then, there is $t_{k,d} \in \mathcal{T}$ such that $t_{k,d} \in T$ and $t_{k,d} \notin H$ for some $0 \leq k < \lambda$ and $0 \leq d \leq \nu$. Without loss of generality let this $k$ be such that for all $0 < k' < k$, we have $t_{k',d} \in T$ iff $t_{k',d} \in H$. Now let us analyze the two cases for $k$.

$\boldsymbol{k = 0}$: Then $d = 0$ since $\tau(0) = 0$ because $\langle T, T\rangle$ is trimmed inducing $\tau$. Then $t_{0,0} \notin H$ which is a contradiction since $\langle H, T\rangle \models t_{0,0}$.

$\boldsymbol{k > 0}$: Then $\tau(k-1) = d'$ with $d' < d$, where $t_{k-1,d'} \in T$ and $t_{k-1,d'} \in H$ since we selected $k$ as the first to not be in the set, therefore, $\langle H, T\rangle \models t_{k-1,d'}$. Since $\langle H, T\rangle \models \Delta_{\lambda,\nu}$ and the head of the rule holds, then $\langle H, T\rangle \models \bigvee_{d' < d'' \leq \nu} t_{k,d''}$. This means $\langle H, T\rangle \models t_{k,d''}$ for some $d' < d'' \leq \nu$, so $t_{k,d''} \in H$, and given that it is timed we have $\tau(k) = d''$. Since $\tau(k) = d$ we have that $d'' = d$, which is a contradiction because $t_{k,d} \notin H$. □

Downloaded from https://www.cambridge.org/core. UNIVERSIDAD AUTONOMA METROPOLITANA, on 08 Nov 2026 at 14:47:57, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://doi.org/10.1017/S1471068426100714

*Lemma 4.*
Let $P$ be a metric logic program and $\mathbf{M} = (\langle \mathbf{H}, \mathbf{T} \rangle, \tau)$ a timed HT-trace of length $\lambda$.
If $\mathbf{M} \models P$, then $\theta(\mathbf{M}) \models \Pi_\lambda(P) \cup \Delta_{\lambda,\nu} \cup \Psi_{\lambda,\nu}(P)$ with $\nu = \tau(\lambda - 1)$.

*Proof.*
We prove that $\theta(\mathbf{M})$ models each part of the translation.

We know $\mathbf{M} \models P$ so due to Corollary 3, we have that $\mathbf{M} \models \Omega(P)$. With the Definition for $\Omega(P)$, get $\mathbf{M} \models \Omega(\Box r)$ for all $\Box r \in P$, which means that $\mathbf{M}, k \models \Box\Omega(r)$ for all $\Box r \in P, 0 \leq k < \lambda$. With Lemma 1, we have that $\theta(\mathbf{M}) \models (\Omega(r))_k$ for all $\Box r \in P, 0 \leq k < \lambda$. Since intervals are irrelevant in $(r)_k$, we then know $(\Omega(r))_k = (r)_k$ and therefore $\theta(\mathbf{M}) \models (r)_k$ for all $\Box r \in P, 0 \leq k < \lambda$, which by definition is $\Pi_\lambda(P)$.

$\theta(\mathbf{M})$ is timed inducing $\tau$ by construction, so it is enough to apply Proposition 2 to show that $\theta(\mathbf{M}) \models \Delta_{\lambda,\nu}$.

Assume towards a contradiction that $\theta(\mathbf{M}) \not\models \Psi_{\lambda,\nu}(P)$. Then, there is a rule $r = \bot \leftarrow (\beta)_k \wedge t_{k,d} \wedge t_{k+1,d'}$ for some $0 \leq k < \lambda - 1$ and $0 \leq d < d' \leq \nu$, such that $\theta(\mathbf{M}) \models (\beta)_k \wedge t_{k,d} \wedge t_{k+1,d'}$, where $\Box(\bigcirc_{[m..n)} a \leftarrow \beta) \in P$. Given that $\theta(\mathbf{M}) \models (\beta)_k$, with Lemma 2 we have $\mathbf{M}, k \models \beta$. Which means that $\mathbf{M}, k \models \bigcirc_{[m..n)} a$ since $\mathbf{M} \models P$. With MHT semantics, we have $\mathbf{M}, k+1 \models a$ and $\tau(k+1) - \tau(k) \in [m..n)$. Given that $\mathbf{M}$ is timed inducing $\tau$, we have $\tau(k) = d$ and $\tau(k+1) = d'$. Then, we reach a contradiction since the conditions for $r$ are either $d - d' > -m$ or $n \in \mathbb{N}$ and $d' - d > n$, for equations (14) and (15) respectively, which are both contradicted by the fact that $\tau(k+1) - \tau(k) \in [m..n)$. □

*Lemma 5.*
Let $P$ be a metric logic program, $\lambda, \nu \in \mathbb{N}$, and $\langle H, T \rangle$ an HT interpretation timed wrt. $\lambda$.
If $\langle H, T \rangle \models \Pi_\lambda(P) \cup \Psi_{\lambda,\nu}(P)$, then $\sigma(\langle H, T \rangle) \models P$.

*Proof.*
Let $\tau$ be the timing function induced by $\langle H, T \rangle$.

Since $\langle H, T \rangle \models \Pi_\lambda(P)$, then $\langle H, T \rangle \models (r)_k$ for all $\Box r \in P, 0 \leq k < \lambda$. We know that $(r)_k = (\Omega(r))_k$ since intervals are not relevant for this definition. So $\langle H, T \rangle \models (\Omega(r))_k$ for all $\Box r \in P, 0 \leq k < \lambda$. With this we can apply Lemma 2 with $\mathbf{M} = \sigma(\langle H, T \rangle)$, to get $\sigma(\langle H, T \rangle), k \models \Omega(r)$ for all $\Box r \in P, 0 \leq k < \lambda$. We now analyze the two possible forms for $r$ and show that $\sigma(\langle H, T \rangle), k \models r$, which would mean that $\sigma(\langle H, T \rangle) \models P$.

$\boldsymbol{r = a_1 \vee \ldots \vee a_m \leftarrow \beta}$: Since there are no metric operators in $r$, we have $\Omega(r) = r$ so we have $\sigma(\langle H, T \rangle), k \models r$

$\boldsymbol{r = \bigcirc_I a \leftarrow \beta}$: If $\sigma(\langle H, T \rangle), k \not\models \beta$ then $\sigma(\langle H, T \rangle), k \models r$ is trivially satisfied. Otherwise, we must show that $\sigma(\langle H, T \rangle), k \models \bigcirc_I a$. We know that $\sigma(\langle H, T \rangle), k \models \bigcirc_{[0..\omega)} a$ because $\langle H, T \rangle \models (\Omega(r))_k$ and $\sigma(\langle H, T \rangle), k \models \beta$, so it is enough to show that $\tau(k+1) - \tau(k) \in I$ to have the MHT semantics for $\bigcirc_I a$. Assume towards a contradiction that $\tau(k+1) - \tau(k) \notin I$, with $I = [m..n)$. Since there are no metric operators in $\beta$, we have $\langle H, T \rangle \models (\beta)_k$ because of Lemma 1. We also know that $\langle H, T \rangle \models t_{k,\tau(k)}$ and $\langle H, T \rangle \models t_{k+1,\tau(k+1)}$, since $\langle H, T \rangle$ induces $\tau$, so $\langle H, T \rangle \models (\beta)_k \wedge t_{k,\tau(k)} \wedge t_{k+1,\tau(k+1)}$. If the reason for $\tau(k+1) - \tau(k) \notin I$ is that $\tau(k+1) - \tau(k) < m$ then $\langle H, T \rangle \models \bot$ because it would meet the conditions of rule equation (14). If the reason is that $n \in \mathbb{N}$ and $\tau(k+1) - \tau(k) \geq n$

Downloaded from https://www.cambridge.org/core. UNIVERSIDAD AUTONOMA METROPOLITANA, on 08 Nov 2026 at 14:47:57, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://doi.org/10.1017/S1471068426100714

then $\langle H, T\rangle \models \bot$ because it would meet the conditions of rule equation (15). In both cases reaching a contradiction, therefore $\tau(k+1) - \tau(k) \in I$. □

*Proposition 1.*
Let $\lambda, \nu \in \mathbb{N}$.

If $\langle T, T\rangle$ is an equilibrium model of $\Delta_{\lambda,\nu}$ then $\langle T, T\rangle$ is timed wrt $\lambda$.

*Proof.*
Let us construct a relation $R = \{(k, d) \mid \langle T, T\rangle \models t_{k,d}\}$.

We proof that $R$ represents a function $\tau : N \to D$ where $0, \ldots, \lambda - 1 \in N$ and $0, \ldots, \nu \in D$. Let us proof by induction that for all $k \in N$ there is exactly one $d$ such that $(k, d) \in R$.

$\boldsymbol{k = 0}$: Since $\langle T, T\rangle \models t_{0,0}$ due to Definition 13, we know $(0, 0) \in R$. We show that there is no other $d$ where $(0, d) \in R$ with $d \neq 0$. Assume towards a contradiction that there is one, then $\langle T, T\rangle \models t_{0,d}$, and $t_{0,d} \in T$ due to HT semantics. We argue that $\langle T \setminus \{t_{0,d}\}, T\rangle \models \Delta_{\lambda,\nu}$. This is the case since $t_{0,d}$ can't appear in the head of rule $\bigvee_{d<d'\leq\nu} t_{k+1,d'} \leftarrow t_{k,d}$. So we can remove $t_{0,d}$ from $T$ and still satisfy $\Delta_{\lambda,\nu}$. Therefore, $\langle T, T\rangle$ is not in equilibrium which is a contradiction.

$\boldsymbol{k + 1 \in [1..\lambda)}$: By induction hypothesis, there is exactly one $d$ such that $(k, d) \in R$, so by construction $\langle T, T\rangle \models t_{k,d}$. With rule equation (13) and HT semantics, we get $\langle T, T\rangle \models \bigvee_{d<d'\leq\nu} t_{k+1,d'}$. This means that $\langle T, T\rangle \models t_{k+1,d'}$ for some $d'$, so by construction $(k + 1, d') \in R$. Now we show that there is no $d'' \neq d'$ such that $(k + 1, d'') \in R$. Assume towards a contradiction there is $(k + 1, d'') \in R$ with $d'' \neq d'$. Then $\langle T, T\rangle \models t_{k+1,d''}$ and $t_{k+1,d''} \in T$. We argue that $\langle T', T\rangle \models \Delta_{\lambda,\nu}$ with $T' = T \setminus \{t_{k+1,d''}\}$. Notice $t_{k+1,d''}$ only appears as a head in rule $\bigvee_{d<d'\leq\nu} t_{k+1,d'} \leftarrow t_{k,d}$. Since $t_{k,d} \in T$, after removing $t_{k+1,d''}$ , $\bigvee_{d<d'\leq\nu} t_{k+1,d'}$ is still satisfied, so $\langle T, T\rangle$ is not in equilibrium, which is a contradiction.

Now we prove that $\tau$ is a (strict) timing function wrt $\lambda$. First, $\tau(0) = 0$ as shown before. We assume towards a contradiction that $\tau(k) \geq \tau(k + 1)$ for some $k$. By construction $(k, d), (k + 1, d') \in R$ with $d \geq d'$, and $\langle T, T\rangle \models t_{k+1,d'}$ and $\langle T, T\rangle \models t_{k,d}$. Then, there must be a supporting rule for $t_{k+1,d'}$. This must be rule $\bigvee_{d<d'\leq\nu} t_{k+1,d'} \leftarrow t_{k,d}$, since it is the only rule that can introduce $t_{k+1,d'}$. Then $d < d'$ to appear in the head, which contradicts the assumption. □

*Proposition 2.*
Let $\langle H, T\rangle$ be an HT interpretation and $\lambda \in \mathbb{N}$.

If $\langle H, T\rangle$ is timed wrt $\lambda$ and induces $\tau$, then $\langle H, T\rangle$ is an HT-model of $\Delta_{\lambda,\nu}$ for $\nu = \tau(\lambda - 1)$.

*Proof.*
We proof that $\langle H, T\rangle$ satisfies every rule in $\Delta_{\lambda,\nu}$

$\boldsymbol{t_{0,0}}$: $\tau$ is a timing function so $\tau(0) = 0$. Since $\langle H, T\rangle$ induces $\tau$ we have that $\langle H, T\rangle \models t_{0,0}$

$\boldsymbol{\bigvee_{d<d'\leq\nu} t_{k+1,d'} \leftarrow t_{k,d}}$: Let$k \in [0..\lambda - 1)$ and $d \in [0..\nu)$. We analyze the two cases for $k$.

$\boldsymbol{\tau(k) \neq d}$: The definition of timed gives us $\langle H, T\rangle \not\models t_{k,d}$ and $\langle T, T\rangle \not\models t_{k,d}$, and with HT semantics we get $\langle H, T\rangle \models \bigvee_{d<d'\leq\nu} t_{k+1,d'} \leftarrow t_{k,d}$, since the body is false.

Downloaded from https://www.cambridge.org/core. UNIVERSIDAD AUTONOMA METROPOLITANA, on 08 Nov 2026 at 14:47:57, subject to the Cambridge Core terms of use , available at https://www.cambridge.org/core/terms. https://doi.org/10.1017/S1471068426100714

$\boldsymbol{\tau(k) = d}$: $\tau(k+1) = d'$ for some $d' > d$ because $\tau$ is a strict timing function and $k+1$ is in the domain. $\langle H, T\rangle$ induces $\tau$ so $\langle H, T\rangle \models t_{k+1,d'}$ and $\langle T, T\rangle \models t_{k+1,d'}$. Due to HT semantics and the fact that $d' \leq \nu$ since $\nu = \tau(\lambda - 1)$, we get $\langle H, T\rangle \models \bigvee_{d<d'\leq\nu} t_{k+1,d'}$ and $\langle T, T\rangle \models \bigvee_{d<d'\leq\nu} t_{k+1,d'}$ which gives us the consequent of the implication. □

*Theorem 1 (Completeness).*
Let $P$ be a plain metric logic program and $\mathbf{M} = (\langle \mathbf{T}, \mathbf{T}\rangle, \tau)$ a total timed HT-trace of length $\lambda$.

If $\mathbf{M}$ is a metric equilibrium model of $P$, then $\theta(\mathbf{M})$ is an equilibrium model of $\Pi_\lambda(P) \cup \Delta_{\lambda,\nu} \cup \Psi_{\lambda,\nu}(P)$ with $\nu = \tau(\lambda - 1)$.

*Proof.*
We first prove that it models the translation $\Pi_\lambda(P) \cup \Delta_{\lambda,\nu} \cup \Psi_{\lambda,\nu}(P)$ using Lemma 4. Now we prove that it is in equilibrium. By construction $\theta(\mathbf{M}) = \langle T, T\rangle$ is total since $\mathbf{M}$ is total. Assume towards a contradiction that there is $\langle H, T\rangle$ such that $H \subset T$ and $\langle H, T\rangle \models \Pi_\lambda(P) \cup \Delta_{\lambda,\nu} \cup \Psi_{\lambda,\nu}(P)$. Then there is $x \in \mathcal{A}^* \cup \mathcal{T}$ such that $x \in T$ and $x \notin H$. Let's analyze each case.

$\boldsymbol{x \in \mathcal{T}}$: We know that $\theta(\mathbf{M})$ is timed by construction, so with Proposition 19 we get $\theta(\mathbf{M})|_\mathcal{T}$ is an equilibrium model of $\Delta_{\lambda,\nu}$, then $\langle H, T\rangle|_\mathcal{T} \not\models \Delta_{\lambda,\nu}$, which contradicts the hypothesis.

$\boldsymbol{x \in \mathcal{A}^*}$: Then $x = a_k$ for some $0 \leq k < \lambda$ so, there is $a_k \in \mathcal{A}^*$ such that $a_k \in T$ and $a_k \notin H$. Notice that then $\langle H, T\rangle$ is timed wrt $\lambda$ inducing $\tau$ and that $\langle H, T\rangle \models \Pi_\lambda(P) \cup \Psi_{\lambda,\nu}(P)$, which means we can apply Lemma 4 to get $\sigma(\langle H, T\rangle) \models P$. This contradicts that $\mathbf{M}$ is an equilibrium model, because $\sigma(\langle H, T\rangle)$ has the same timing function and it is smaller due to $x$. □

*Theorem 2 (Correctness).*
Let $P$ be a plain metric logic program, and $\lambda, \nu \in \mathbb{N}$.

If $\langle T, T\rangle$ is an equilibrium model of $\Pi_\lambda(P) \cup \Delta_{\lambda,\nu} \cup \Psi_{\lambda,\nu}(P)$, then $\sigma(\langle T, T\rangle)$ is a metric equilibrium model of $P$.

*Proof.*
We argue that $\langle T, T\rangle$ is an equilibrium model of $\Delta_{\lambda,\nu}$ since this program is over alphabet $\mathcal{T}$ and $\Pi_\lambda(P)$ is over alphabet $\mathcal{A}^*$, where $\mathcal{T} \cap \mathcal{A}^* = \emptyset$. Additionally $\Psi_{\lambda,\nu}(P)$ consist of only integrity constraints. We can apply the splitting property (Lemma 3) to say that $\langle T, T\rangle|_\mathcal{T}$ is an equilibrium model of $\Delta_{\lambda,\nu}$, $\langle T, T\rangle|_\mathcal{T}$ is timed wrt. $\lambda$ inducing timing function $\tau$ due to Proposition 1. Notice that $\langle T, T\rangle$ is also timed. Knowing this, we have the necessary conditions to create $\sigma(\langle T, T\rangle)$. Then, we can apply Lemma 5 to get that $\sigma(\langle T, T\rangle) \models P$.

We now prove that $\sigma(\langle T, T\rangle)$ is an equilibrium model. Let $\sigma(\langle T, T\rangle) = (\langle T_k, T_k\rangle_{k\in[0..\lambda)}, \tau)$. First notice that $\sigma(\langle T, T\rangle)$ is total by construction. Let's assume towards a contradiction that $\sigma(\langle T, T\rangle)$ is not minimal. Then, there is some metric trace $\mathbf{M}' = (\langle H_k, T_k\rangle_{k\in[0..\lambda)}, \tau)$ such that $\mathbf{M}' \models P$ and $H_k \subset T_k$ for some $k \in [0..\lambda)$. So, there is $a \in \mathcal{A}$ such that $a \in T_k$ and $a \notin H_k$. By construction $a_k \in T$ and $a_k \notin H$ where $\theta(\mathbf{M}') = \langle H, T\rangle$. We apply Lemma 4 and get $\langle H, T\rangle \models \Pi_\lambda(P) \cup \Delta_{\lambda,\nu} \cup \Psi_{\lambda,\nu}(P)$ so $\langle T, T\rangle$ is not an equilibrium model, which is a contradiction. □

Downloaded from https://www.cambridge.org/core. UNIVERSIDAD AUTONOMA METROPOLITANA, on 08 Nov 2026 at 14:47:57, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://doi.org/10.1017/S1471068426100714

### *C.1.2 Translation of plain metric logic programs to HT$_c$*

*Lemma 6 (Splitting Plain HT$_c$).*
Given a metric logic program $P$ and a $\lambda \in \mathbb{N}$, the stable models of $\Pi_\lambda(P) \cup \Delta^c_\lambda \cup \Psi^c_\lambda(P)$ can be computed by means of the splitting technique.

*Proof.*
Proven as Lemma 3 but using the notion of splitting set as defined in Cabalar et al. (2020c, Definition 10). □

*Proposition 20 (Model if timed equilibrium).*
Let $\langle h, t\rangle$ be an HT$_c$ interpretation and $\lambda \in \mathbb{N}$.

If $\langle h, t\rangle$ is timed wrt. $\lambda$ and induces $\tau$ then $\langle h, t\rangle|_{\mathcal{T}^c}$ is a total HT$_c$ interpretation and an constraint equilibrium model of $\Delta^c_\lambda$.

*Proof.*
The satisfaction follows from Proposition 5. From the definition of being timed it follows that $\langle h, t\rangle|_{\mathcal{T}^c} = \langle t', t'\rangle$ must be total. Now we prove that it is in equilibrium. Assume towards a contradiction that there is $\langle h', t'\rangle$ such that $h' \subset t'$ and $\langle h', t'\rangle \models \Delta^c_\lambda$ Then, there is $(t_k, d)\int'$ such that $(t_k, d) \notin h'$ for some $0 \leq k < \lambda$ and $d \in \mathbb{N}$, so $h'(t_k) = \boldsymbol{u}$. We analyze the two cases for $k$.

$\boldsymbol{k = 0}$: Then $h' \notin [\![\, t_0 = 0 \,]\!]$ since $h'(t_0) = \boldsymbol{u}$. Contradiction since $\langle h', t'\rangle \models t_0 = 0$ because $\langle h', t'\rangle \models \Delta^c_\lambda$.

$\boldsymbol{k > 0}$: Since $h'(t_k) = \boldsymbol{u}$ and $x \in \mathbb{N}$ is a condition for $[\![\, x - y \leq d \,]\!]$, we have $h' \notin [\![\, t_k - t_{k+1} \leq -1 \,]\!]$. Contradiction since $\langle h', t'\rangle \models t_k - t_{k+1} \leq -1$ because $\langle h', t'\rangle \models \Delta^c_\lambda$. □

*Lemma 7 (Correctness HT$_c$).*
Let $P$ be a metric logic program, $\langle h, t\rangle$ an HT$_c$ interpretation timed wrt. $\lambda$.

If $\langle h, t\rangle \models \Pi_\lambda(P) \cup \Psi^c_\lambda(P)$, then $\sigma^c(\langle h, t\rangle) \models P$.

*Proof.*
Let $\tau$ be the timing function induced by $\langle h, t\rangle$. We will prove that $\sigma^c(\langle h, t\rangle) \models P$ by showing that it models each rule $r \in P$. For this, first notice that $\langle h, t\rangle \models \Pi_\lambda(P)$, so $\langle h, t\rangle \models (r)_k$ for all $\Box r \in P, 0 \leq k < \lambda$. Since $(r)_k = (\Omega(r))_k$, then $\langle h, t\rangle \models (\Omega(r))_k$, therefore we can apply Lemma 2 to get $\sigma^c(\langle h, t\rangle), k \models \Omega(r)$. Notice that Lemma 1 is defined for HT, but since it uses only boolean variables it holds the for HT$_c$ (Observation 1 in Cabalar *et al.* (2016)). Now we analyze the two cases for $r$ and show that $\sigma^c(\langle h, t\rangle), k \models r$, which would mean that $\sigma^c(\langle h, t\rangle) \models P$.

$\boldsymbol{r = a_1 \vee \ldots \vee a_m \leftarrow \beta}$: Since there are no metric operators in $r$, then $\Omega(r) = r$. Therefore, $\sigma^c(\langle h, t\rangle), k \models r$.

$\boldsymbol{r = \bigcirc_I a \leftarrow \beta}$: If $\sigma^c(\langle h, t\rangle), k \not\models \beta$ then $\sigma^c(\langle h, t\rangle), k \models r$ is trivially satisfied. Otherwise, we must show that $\sigma^c(\langle h, t\rangle), k \models \bigcirc_I a$. We know that $\sigma^c(\langle h, t\rangle), k \models \bigcirc_{[0..\omega)} a$ because $\langle h, t\rangle \models (\Omega(r))_k$ and $\sigma^c(\langle h, t\rangle), k \models \beta$, so it is enough to show that $\tau(k+1) - \tau(k) \in I$ to have the MHT semantics for $\bigcirc_I a$.

Downloaded from https://www.cambridge.org/core. UNIVERSIDAD AUTONOMA METROPOLITANA, on 08 Nov 2026 at 14:47:57, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://doi.org/10.1017/S1471068426100714

Assume towards a contradiction that $\tau(k+1)-\tau(k) \notin I$, with $I=[m..n)$. Since there are no metric operators in $\beta$, we have $\langle h,t\rangle \models (\beta)_k$ because of Lemma 1. If the reason for $\tau(k+1)-\tau(k) \notin I$ is that $\tau(k+1)-\tau(k) < m$ then $t_{k+1}-t_k < m$, since it is timed inducing $\tau$. This is equivalent to $t_k - t_{k+1} > -m$ so $h \notin [\![\, t_k - t_{k+1} \leq -m \,]\!]$ and $h \models \neg(t_k - t_{k+1} \leq -m)$. Therefore, $\langle H,T\rangle \models \bot$ because of rule equation (19), which is a contradiction. If the reason is that $n \in \mathbb{N}$ and $\tau(k+1)-\tau(k) \geq n$, then we reach a contradiction by reasoning analogously and using rule equation (20). Therefore, $\tau(k+1)-\tau(k) \in I$. □

*Lemma 8 (Completeness HT$_c$).*
Let $P$ be a metric logic program and $\mathbf{M}=(\langle \mathbf{H},\mathbf{T}\rangle,\tau)$ a timed HT-trace of length $\lambda$.
If $\mathbf{M} \models P$, then $\theta^c(\mathbf{M}) \models \Pi_\lambda(P) \cup \Delta^c_\lambda \cup \Psi^c_\lambda(P)$.

*Proof.*
We prove that $\theta^c(\mathbf{M})$ models each part of the translation.

We know $\mathbf{M} \models P$ so due to Corollary 3,[13] we have that $\mathbf{M} \models \Omega(P)$. With the Definition for $\Omega(P)$, get $\mathbf{M} \models \Omega(\Box r)$ for all $\Box r \in P$, which means that $\mathbf{M},k \models \Box\Omega(r)$ for all $\Box r \in P, 0 \leq k < \lambda$. With Lemma 1,[14] we have that $\theta^c(\mathbf{M}) \models (\Omega(r))_k$ for all $\Box r \in P, 0 \leq k < \lambda$. Since intervals are irrelevant in $(r)_k$, we then know $(\Omega(r))_k = (r)_k$ and therefore $\theta^c(\mathbf{M}) \models (r)_k$ for all $\Box r \in P, 0 \leq k < \lambda$, which by definition is $\Pi_\lambda(P)$.

$\theta^c(\mathbf{M})$ is timed inducing $\tau$ by construction, so it is enough to apply Proposition 5 to show that $\theta^c(\mathbf{M}) \models \Delta^c_\lambda$.

Assume towards a contradiction that $\theta^c(\mathbf{M}) \not\models \Psi^c_\lambda(P)$. Then, there is a rule $r \in \Psi^c_\lambda(P)$ such that $\theta^c(\mathbf{M}) \not\models r$. Without loss of generality, let $r$ come from equation (19), where the case for equation (20) is analogous. Then, $r = \bot \leftarrow (\beta)_k \wedge \neg(t_k - t_{k+1} \leq -m)$ for some $0 \leq k < \lambda - 1$ and $\theta^c(\mathbf{M}) \models (\beta)_k \wedge \neg(t_k - t_{k+1} \leq -m)$, where $\Box(\bigcirc_{[m..n)} a \leftarrow \beta) \in P$. Given that $\theta^c(\mathbf{M}) \models (\beta)_k$, with Lemma 2 we have $\mathbf{M},k \models \beta$. Which means that $\mathbf{M},k \models \bigcirc_{[m..n)} a$ since $\mathbf{M} \models P$. With MHT semantics, we have $\mathbf{M},k+1 \models a$ and $\tau(k+1)-\tau(k) \in [m..n)$. Given that $\mathbf{M}$ is timed inducing $\tau$, we have $\tau(k) = t_k$ and $\tau(k+1) = t_k$, so $t_{k+1} - t_k \geq m$, which is equivalent to $t_k - t_{k+1} \leq -m$. Then $h \in [\![\, t_k - t_{k+1} \leq -m \,]\!]$ and $\theta^c(\mathbf{M}) \models (t_k - t_{k+1} \leq -m)$, which is a contradiction, since $\theta^c(\mathbf{M}) \not\models (t_k - t_{k+1} \leq -m)$. □

*Proposition 3.*
Let $\langle h,t\rangle$ be an HT$_c$ interpretation and $\lambda \in \mathbb{N}$
If $\langle h,t\rangle$ is an HT$_c$-model of $\Delta^c_\lambda$, then $h(t_k) \in \mathbb{N}$ for all $0 \leq k < \lambda$.

*Proof.*
Case analysis on $k$

$\boldsymbol{k=0}$: Since $t_0 = 0 \in \Delta^c_\lambda$ then $h \in [\![\, t_0 = 0 \,]\!]$. Therefore, $h(t_0) \in \mathbb{N}$, since it is a condition of the denotation.

$\boldsymbol{0 < k < \lambda}$: Since $t_{k-1} - t_k \leq -1 \in \Delta^c_\lambda$, then $h \in [\![\, t_{k-1} - t_k \leq -1 \,]\!]$. Therefore, $h(t_k) \in \mathbb{N}$ since it is a condition of the denotation. □

[13] Notice that Corollary 3 is defined for HT, but since it uses only boolean variables it holds the for HT$_c$ (Observation 1 in Cabalar *et al.* (2016)).
[14] Notice that Lemma 1 is defined for HT, but since it uses only boolean variables it holds the for HT$_c$ (Observation 1 in Cabalar *et al.* (2016)).

Downloaded from https://www.cambridge.org/core. UNIVERSIDAD AUTONOMA METROPOLITANA, on 08 Nov 2026 at 14:47:57, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://doi.org/10.1017/S1471068426100714

*Proposition 4.*
Let $\langle h, t\rangle$ be an $\mathrm{HT}_c$ interpretation and $\lambda \in \mathbb{N}$.

If $\langle h, t\rangle$ is an $\mathrm{HT}_c$-model of $\Delta_\lambda^c$ then $\langle h, t\rangle$ is timed wrt $\lambda$.

*Proof.*
Let us construct the induced timing function $\tau$ such that $\tau(k) = h(t_k)$ for $0 \leq k \leq \lambda$. We prove that $\tau$ is a timing function by showing that it is strictly increasing. Notice that the functional nature of $t_k$ is already given by the $\mathrm{HT}_c$ semantics, unlike for the HT approach. First, $\tau(0) = 0$ is implied by the denotation since $\langle h, t\rangle \models t_0 = 0$. To show that it is increasing, let $k \in [0..\lambda - 1)$. We have that $\langle h, t\rangle \models t_k - t_{k+1} \leq -1$. This means that $h(t_k) < h(t_{k+1})$ by the denotation semantics. Therefore $\tau(k) < \tau(k+1)$ by construction. □

*Proposition 5.*
Let $\langle h, t\rangle$ be an $\mathrm{HT}_c$ interpretation and $\lambda \in \mathbb{N}$.

If $\langle h, t\rangle$ is timed wrt $\lambda$ then $\langle h, t\rangle$ is an $\mathrm{HT}_c$-model of $\Delta_\lambda^c$.

*Proof.*
Let $\tau$ be the timing function induced by $\langle h, t\rangle$. We prove that $\langle h, t\rangle$ models each rule in $\Delta_\lambda^c$. Since $\tau(0) = 0$ then $h(t_0) = 0$, because $\tau$ was induced by $\langle h, t\rangle$. Therefore, $\langle h, t\rangle \models t_0 = 0$. Let $0 \leq k < \lambda - 1$. Since $\tau$ is a timing function, $\tau(k) < \tau(k+1)$. This means that $h(t_k) < h(t_{k+1})$. Therefore, $\langle h, t\rangle \models t_k - t_{k+1} \leq -1$. □

*Theorem 3 (Completeness).*
Let $P$ be a plain metric logic program and $\mathbf{M} = (\langle \mathbf{T}, \mathbf{T}\rangle, \tau)$ a total timed HT-trace of length $\lambda$.

If $\mathbf{M}$ is a metric equilibrium model of $P$, then $\theta^c(\mathbf{M})$ is a constraint equilibrium model of $\Pi_\lambda(P) \cup \Delta_\lambda^c \cup \Psi_\lambda^c(P)$.

*Proof.*
We first prove that it models the program by applying Lemma 4. Now we prove that it is in equilibrium. By construction $\theta^c(\mathbf{M}) = \langle t, t\rangle$ is total since $\mathbf{M}$ is total. Assume towards a contradiction that there is $\langle h, t\rangle$ such that $h \subset t$ and $\langle h, t\rangle \models \Pi_\lambda(P) \cup \Delta_\lambda^c \cup \Psi_\lambda^c(P)$. Then, there is some $(x, v)$ such that $(x, v) \int$ and $(x, v) \notin h$. We analyze the two cases for $x$.

$\boldsymbol{x = t_k}$: If $(t_k, v) \notin h$ then there is no assignment for $t_k$ because $h \subset t$. This contradict with Proposition 3 since $\langle h, t\rangle \models \Delta_\lambda^c$

$\boldsymbol{x = a_k \in \mathcal{A}^*}$: We apply Lemma 7 and get $\sigma(\langle h, t\rangle) \models P$, where $\sigma(\langle h, t\rangle) = (\langle \mathbf{H}, \mathbf{T}\rangle, \tau)$. Notice that $\mathbf{T}$ and $\tau$ are the same as in $\mathbf{M}$ since $\langle h, t\rangle$ induces $\tau$ and that $\mathbf{H} < \mathbf{T}$ since $a \in T_k$ but $a \notin H_k$. This is a contradiction since $\mathbf{M}$ is in equilibrium. □

*Theorem 4 (Correctness).*
Let $P$ be a plain metric logic program, and $\lambda \in \mathbb{N}$.

If $\langle t, t\rangle$ is a constraint equilibrium model of $\Pi_\lambda(P) \cup \Delta_\lambda^c \cup \Psi_\lambda^c(P)$, then $\sigma^c(\langle t, t\rangle)$ is a metric equilibrium model of $P$.

*Proof.*
We can prove that $\sigma^c(\langle t, t\rangle) \models P$ by applying Lemma 7. For this we only need to show that $\langle t, t\rangle$ is timed, which follows from Proposition 4, given that $\langle t, t\rangle \models \Delta_\lambda^c$. We it remains

Downloaded from https://www.cambridge.org/core. UNIVERSIDAD AUTONOMA METROPOLITANA, on 08 Nov 2026 at 14:47:57, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://doi.org/10.1017/S1471068426100714

to prove that $\sigma^c(\langle t,t\rangle)$ is an equilibrium model. Let $\sigma^c(\langle t,t\rangle) = (\langle T_k, T_k\rangle_{k\in[0..\lambda)}, \tau)$, and let us assume towards a contradiction that $\sigma^c(\langle t,t\rangle)$ is not minimal. There is some metric trace $\mathbf{M}' = (\langle H_k, T_k\rangle_{k\in[0..\lambda)}, \tau)$ such that $\mathbf{M}' \models P$ and $H_k \subset T_k$ for some $k \in [0..\lambda)$. This means, that there is $a \in \mathcal{A}$ such that $a \in T_k$ and $a \notin H_k$, so $(a_k, \boldsymbol{t}) \in t$ and $(a_k, \boldsymbol{t}) \notin h$ where $\theta(\mathbf{M}') = \langle h, t\rangle$. By applying Lemma 4 we get that $\langle h, t\rangle \models \Pi_\lambda(P) \cup \Delta^c_\lambda \cup \Psi^c_\lambda(P)$. This is a contradiction since $\langle t,t\rangle$ is a constraint equilibrium model. □

### C.2 General metric logic programs

*Definition 16 (Interval alphabet).*
We define the interval alphabet

$$\mathfrak{I} = \{\delta_I^{k,j} \mid I \in \mathbb{I}, k, j \in \mathbb{N}, k \leq j\} \tag{C1}$$

*Definition 17 (Auxiliary alphabet).*
Given a metric logic program $P$, we define the alphabet

$$\mathcal{L}_P = \{L^k_\mu \mid \mu \in P, k \in \mathbb{N}\} \cup \{L^{k,j}_{\odot_I b} \mid \odot \in \{\Diamond, \Box\}, \odot_I b \in P, k, j \in \mathbb{N}\} \tag{C2}$$

*Proposition 21 (Equivalence of auxiliary atoms).*
Given $\lambda \in \mathbb{N}$, a (simple) metric formula $\mu$ over $\mathcal{A}$, an HT interpretation $\langle H, T\rangle$ with its corresponding three-valued interpretation $\boldsymbol{m}$ and a timed trace $\mathbf{M} = (\langle \mathbf{H}, \mathbf{T}\rangle, \tau)$ of length $\lambda$ with its corresponding metric three-valued interpretation $\boldsymbol{m}^\tau$, such that the following propositions hold:

e1. $\boldsymbol{m}(a_k) = \boldsymbol{m}^\tau(k, a)$ for all $k \in [0..\lambda)$, $a \in \mathcal{A}$
e2. $\boldsymbol{m}(\delta_I^{k,j}) = 2$ iff $\tau(j) - \tau(k) \notin I$ for all $k \in [0..\lambda)$, $I \in \mathbb{I}$
e3. $\boldsymbol{m}(\delta_I^{k,j}) = 0$ iff $\tau(j) - \tau(k) \in I$ for all $k \in [0..\lambda)$, $I \in \mathbb{I}$
e4. if $\mu = \Box_I b$ then $\boldsymbol{m}(L^{k,j}_\mu) = \max\{\boldsymbol{m}(L^j_b), \boldsymbol{m}(\delta_I^{k,j})\}$ for all $k \in [0..\lambda)$, $j \in [k..\lambda)$
e5. if $\mu = \Diamond_I b$ then $\boldsymbol{m}(L^{k,j}_\mu) = \min\{\boldsymbol{m}(L^j_b), \boldsymbol{m}(\neg\delta_I^{k,j})\}$ for all $k \in [0..\lambda)$, $j \in [k..\lambda)$

Then, the following holds for any $k \in [0..\lambda)$:

$$\boldsymbol{m}(\eta_k(\mu)) = 2 \text{ iff } \boldsymbol{m}(L^k_\mu) = \boldsymbol{m}^\tau(k, \mu) \tag{C3}$$

*Proof.*
We proceed by case analysis on the structure of $\mu$.

$\boldsymbol{\mu = a \in \mathcal{A}}$: We start with $\boldsymbol{m}(L^k_a) = \boldsymbol{m}^\tau(k, a)$. Since condition e1 holds, we have that $\boldsymbol{m}(L^k_a) = \boldsymbol{m}(a_k)$. By applying Proposition 8, we get that $\boldsymbol{m}(L^k_a \leftrightarrow a_k) = 2$.

$\boldsymbol{\mu = \bot}$: We start with $\boldsymbol{m}(L^k_\bot) = \boldsymbol{m}^\tau(k, \bot)$. Using the semantics of the metric three-valued interpretation, we have that $\boldsymbol{m}(L^k_\bot) = 0$. With the semantics of HT three valued we get that $\boldsymbol{m}(L^k_\bot) = \boldsymbol{m}(\bot)$. By applying Proposition 8, we get that $\boldsymbol{m}(L^k_\bot \leftrightarrow \bot) = 2$.

$\boldsymbol{\mu = \top}$: Analogous to the previous case.

$\boldsymbol{\mu = \mathbf{F}}$: We start with $\boldsymbol{m}(L^k_\mathbf{F}) = \boldsymbol{m}^\tau(k, \mathbf{F})$. Now we analyze the two cases for $k$.

$\boldsymbol{k < \lambda - 1}$: With the semantics of the metric three-valued interpretation, we have that $\boldsymbol{m}(L^k_\mathbf{F}) = 0$. With the semantics of HT three valued we get that $\boldsymbol{m}(L^k_\mathbf{F}) = \boldsymbol{m}(\mathbf{F})$. By applying Proposition 8, we get that $\boldsymbol{m}(L^k_\mathbf{F} \leftrightarrow \mathbf{F}) = 2$.

Downloaded from https://www.cambridge.org/core. UNIVERSIDAD AUTONOMA METROPOLITANA, on 08 Nov 2026 at 14:47:57, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://doi.org/10.1017/S1471068426100714

$\boldsymbol{k = \lambda - 1}$: With the semantics of the metric three-valued interpretation, we have that $\boldsymbol{m}(L^k_{\mathbf{F}}) = 1$. With the semantics of HT three valued we get that $\boldsymbol{m}(L^k_{\mathbf{F}}) = \boldsymbol{m}(\mathbf{F})$. By applying Proposition 8, we get that $\boldsymbol{m}(L^k_{\mathbf{F}} \leftrightarrow \mathbf{F}) = 2$.

$\boldsymbol{\mu = \mathbf{I}}$: Analogous to the previous case.

$\boldsymbol{\mu = \bigcirc_I b}$: We start with $\boldsymbol{m}(L^k_{\bigcirc_I b}) = \boldsymbol{m}^\tau(k, \bigcirc_I b)$, and analyze the two cases for $k$.

$\boldsymbol{k < \lambda - 1}$: Now further analysis on wheather a one step falls in the interval.

$\boldsymbol{\tau(k+1) - \tau(k) \notin I}$: Three-valued semantics for metric gives us $\boldsymbol{m}(L^k_{\bigcirc_I b}) = 0$, which is equivalent to $\min\{0, \boldsymbol{m}(L^{k+1}_b)\}$. Using condition e2. we get $\min\{\boldsymbol{m}(\neg\delta^{k,j}_I), \boldsymbol{m}(L^{k+1}_b)\}$, since $\boldsymbol{m}(\neg\delta^{k,j}_I) = 0$ because $\boldsymbol{m}(\delta^{k,j}_I) = 2$. Finally $\boldsymbol{m}(L^k_{\bigcirc_I b}) = \boldsymbol{m}(\neg\delta^{k,j}_I \wedge L^{k+1}_b)$. Therefore, $\boldsymbol{m}(L^k_{\bigcirc_I b} \leftrightarrow \neg\delta^{k,j}_I \wedge L^{k+1}_b) = 2$, by applying Proposition 8.

$\boldsymbol{\tau(k+1) - \tau(k) \in I}$: Three-valued semantics for metric gives us $\boldsymbol{m}(L^k_{\bigcirc_I b}) = \boldsymbol{m}^\tau(k+1, b)$. Due to condition e1. we have that $\boldsymbol{m}^\tau(k+1, b) = \boldsymbol{m}(L^{k+1}_b)$, so $\boldsymbol{m}(L^k_{\bigcirc_I b}) = \boldsymbol{m}(L^{k+1}_b)$, which is equivalent to $\min\{2, \boldsymbol{m}(L^{k+1}_b)\}$. Using condition e3. we get $\min\{\boldsymbol{m}(\neg\delta^{k,j}_I), \boldsymbol{m}(L^{k+1}_b)\}$ since $\boldsymbol{m}(\neg\delta^{k,j}_I) = 2$, because $\boldsymbol{m}(\delta^{k,j}_I) = 0$. Finally $= \boldsymbol{m}(\neg\delta^{k,j}_I \wedge L^{k+1}_b)$. Therefore, $\boldsymbol{m}(L^k_{\bigcirc_I b} \leftrightarrow \neg\delta^{k,j}_I \wedge L^{k+1}_b) = 2$, by applying Proposition 8.

$\boldsymbol{k = \lambda - 1}$: Three-valued semantics for metric gives us $\boldsymbol{m}(L^k_{\bigcirc_I b}) = 0$, which is equivalent to $\boldsymbol{m}(\bot)$. We apply Proposition 8 and get that $\boldsymbol{m}(L^k_{\bigcirc_I b} \leftrightarrow \bot) = 2$.

$\boldsymbol{\mu = \Box_I b}$: We check that the two equivalences in $\eta_k(\Box_I\, b)$ hold. First, notice that $L^{k,j}_{\Box_I\, b} \leftrightarrow \delta^{k,j}_I \vee L^j_b$ holds trivially by construction. Now for $L^k_{\Box_I\, b} \leftrightarrow \bigwedge_{j=k}^{\lambda-1} L^{k,j}_{\Box_I\, b}$, we start with $\boldsymbol{m}(L^k_{\Box_I}) = \boldsymbol{m}^\tau(k, \Box_I\, b)$. This is equivalent to $\min\{2\} \cup \{\boldsymbol{m}^\tau(j, b) \mid k \le j < \lambda \text{ and } \tau(j) - \tau(k) \in I\}$, which due to condition e1. is equivalent to $\min\{2\} \cup \{\boldsymbol{m}(L^j_b) \mid k \le j < \lambda \text{ and } \tau(j) - \tau(k) \in I\}$. The invariant of min let us us add multiple instances of 2, giving us $\min\{\boldsymbol{m}(L^j_b) \mid k \le j < \lambda \text{ and } \tau(j) - \tau(k) \in I\} \cup \{2 \mid k \le j < \lambda \text{ and } \tau(j) - \tau(k) \notin I\}$. With conditions e2. and e3. we know that the value of $\boldsymbol{m}(\delta^{k,j}_I)$ for the conditions of each set is 0 and 2 respectively, so we can substitute them and get $\min\{\max\{\boldsymbol{m}(L^j_b), \boldsymbol{m}(\delta^{k,j}_I)\} \mid k \le j < \lambda \text{ and } \tau(j) - \tau(k) \in I\} \cup \{\boldsymbol{m}(\delta^{k,j}_I) \mid k \le j < \lambda \text{ and } \tau(j) - \tau(k) \notin I\}$. Now the invariant of max gives us $\min\{\max\{\boldsymbol{m}(L^j_b), \boldsymbol{m}(\delta^{k,j}_I)\} \mid k \le j < \lambda \text{ and } \tau(j) - \tau(k) \in I\} \cup \{\max\{\boldsymbol{m}(L^j_b), \boldsymbol{m}(\delta^{k,j}_I)\} \mid k \le j < \lambda \text{ and } \tau(j) - \tau(k) \notin I\}$. With simple set operations since the values are the same we can make the union and remove the condition $\min\{\max\{\boldsymbol{m}(L^j_b), \boldsymbol{m}(\delta^{k,j}_I)\} \mid k \le j < \lambda\}$, which thanks to condition e4. is equivalent to $\min\{\boldsymbol{m}(L^{k,j}_{\Box_I\, b}) \mid k \le j < \lambda\}$ and finally to $\boldsymbol{m}(\bigwedge_{j=k}^{\lambda-1} L^{k,j}_{\Box_I\, b})$. Therefore $L^k_{\Box_I\, b} \leftrightarrow \bigwedge_{j=k}^{\lambda-1} L^{k,j}_{\Box_I\, b}$

$\boldsymbol{\mu = \Diamond_I b}$: Analogous to the previous case. □

### *C.2.1 Translation of MHT to general metric logic programs*

*Lemma 9 (Completeness).*
Let $\varphi$ be a metric formula over alphabet $\mathcal{A}$ and let $\mathbf{M} = (\langle \mathbf{H}, \mathbf{T} \rangle, \tau)$ be a timed trace of length $\lambda$ with corresponding metric three-valued interpretation $\boldsymbol{m}^\tau$, such that $\mathbf{M} \models \varphi$.

Downloaded from https://www.cambridge.org/core. UNIVERSIDAD AUTONOMA METROPOLITANA, on 08 Nov 2026 at 14:47:57, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://doi.org/10.1017/S1471068426100714

Then, there exists some timed trace $\mathbf{M}' = (\langle \mathbf{H}', \mathbf{T}' \rangle, \tau)$ over alphabet $\mathcal{A} \cup \mathcal{L}_\varphi$ such that $\langle \mathbf{H}, \mathbf{T} \rangle = \langle \mathbf{H}', \mathbf{T}' \rangle|_{\mathcal{A}}$ and $\mathbf{M}' \models \Theta(\varphi)$.

*Proof.*
Given $\boldsymbol{m}^\tau$, we construct $\mathbf{M}'$ by defining its corresponding three-valued metric interpretation $\boldsymbol{m}^{\tau\prime}$ such that $\boldsymbol{m}^{\tau\prime}(k, L_\phi) = \boldsymbol{m}^\tau(k, \phi)$ or any $\phi \in sub(\varphi)$ and $k \in [0..\lambda)$, and for any atom $a \in \mathcal{A}$ we have $\boldsymbol{m}^{\tau\prime}(k, a) = \boldsymbol{m}^\tau(k, a)$ for $k \in [0..\lambda)$. By construction we then know that $\langle \mathbf{H}, \mathbf{T} \rangle = \langle \mathbf{H}', \mathbf{T}' \rangle|_{\mathcal{A}}$.

Now we proceed to proof that $\mathbf{M}' \models \Theta(\varphi)$, so $\mathbf{M}' \models \{\Box \mathbf{I} \to L_\varphi\} \cup \{\eta^*(\phi) \mid \phi \in sub(\varphi)\}$. We can prove $\mathbf{M}' \models \eta(\phi)$ for any $\phi \in sub(\varphi)$ by analyzing the structure as done in the proof for Lemma 10, (note that we use the construction of $\mathbf{M}'$ instead of the induction hypothesis). Now we must show that $\mathbf{M}' \models \{\Box \mathbf{I} \to L_\varphi\}$. We know $\mathbf{M} \models \varphi$ by hypothesis, so $\mathbf{M}, 0 \models \varphi$. Proposition 17 gives us $\boldsymbol{m}^\tau(0, \varphi) = 2$, which by construction of $\mathbf{M}'$ gives us $\boldsymbol{m}^{\tau\prime}(0, L_\varphi) = 2$. Since $\boldsymbol{m}^{\tau\prime}(0, \mathbf{I} \to L_\varphi) = \boldsymbol{m}^{\tau\prime}(0, L_\varphi)$ we have $\boldsymbol{m}^{\tau\prime}(0, \mathbf{I} \to L_\varphi) = 2$. Notice also that $\boldsymbol{m}^{\tau\prime}(j, \mathbf{I} \to L_\varphi) = 2$ for all $j > 0$. since $\boldsymbol{m}^{\tau\prime}(j, \mathbf{I}) = 0$. Then $\min\{2\} \cup \{\boldsymbol{m}^\tau(j, \varphi) \mid 0 \leq j < \lambda \text{ and } \tau(j) - \tau(k) \in I\} = 2$ so $\boldsymbol{m}^{\tau\prime}(0, \Box \mathbf{I} \to L_\varphi) = 2$, which by proposition 17 gives us $\mathbf{M}', 0 \models \Box \mathbf{I} \to L_\varphi$ and $\mathbf{M}' \models \Theta(\varphi)$. □

*Lemma 10 (Correctness).*
Let $\varphi$ be a metric formula over $\mathcal{A}$ and let $\mathbf{M} = (\langle \mathbf{H}, \mathbf{T} \rangle, \tau)$ be a timed trace of $\Theta(\varphi)$, with corresponding metric three-valued interpretation $\boldsymbol{m}^\tau$. Then, for any $\gamma \in sub(\varphi)$ and $k \in [0..\lambda)$, we have $\boldsymbol{m}^\tau(k, L_\gamma) = \boldsymbol{m}^\tau(k, \gamma)$.

*Proof.*
We proceed by structural induction on $\phi$.

$\boldsymbol{\phi \in \{a, \bot, \mathbf{I}\}}$: From equation (24), since $\mathbf{M} \models \Theta(\varphi)$, we obtain $\mathbf{M} \models \Box(L_\phi \leftrightarrow \phi)$. The satisfaction of MHT then ensures that for any $k \in \lambda$, we have $\mathbf{M}, k \models L_\phi \leftrightarrow \phi$. Applying Proposition 17, we conclude that for any $k \in \lambda$, the valuation $\boldsymbol{m}^\tau(k, L_\phi)$ is equal to $\boldsymbol{m}^\tau(k, \phi)$.

$\boldsymbol{\phi = \zeta \otimes \psi}$: From equation (24), we have $\mathbf{M} \models \Box(L_\phi \leftrightarrow L_\zeta \otimes L_\psi)$. From MEL satisfaction, it follows that for any $k \in \lambda$, $\mathbf{M}, k \models L_\phi \leftrightarrow L_\zeta \otimes L_\psi$. Using Proposition 17, we get $\boldsymbol{m}^\tau(k, L_\phi) = \boldsymbol{m}^\tau(k, L_\zeta \otimes L_\psi)$. By Definition 9, this is equal to $f^{\otimes}\{\boldsymbol{m}^\tau(k, L_\zeta), \boldsymbol{m}^\tau(k, L_\psi)\}$. Applying the induction hypothesis, we replace $\boldsymbol{m}^\tau(k, L_\zeta)$ and $\boldsymbol{m}^\tau(k, L_\psi)$ with $\boldsymbol{m}^\tau(k, \zeta)$ and $\boldsymbol{m}^\tau(k, \psi)$, respectively. This simplifies to $\boldsymbol{m}^\tau(k, \phi)$ by Definition 9, completing this case.

$\boldsymbol{\phi = \zeta \to \psi}$: From equation (24), we obtain $\mathbf{M} \models \Box(L_\phi \leftrightarrow L_\zeta \to L_\psi)$. By MEL satisfaction, this implies that for any $k \in \lambda$, $\mathbf{M}, k \models L_\phi \leftrightarrow L_\zeta \to L_\psi$. From Proposition 17, we conclude that $\boldsymbol{m}^\tau(k, L_\phi) = \boldsymbol{m}^\tau(k, L_\zeta \to L_\psi)$.
We now distinguish two cases. If $\boldsymbol{m}^\tau(k, L_\zeta) \leq \boldsymbol{m}^\tau(k, L_\psi)$, then by Definition 9, $\boldsymbol{m}^\tau(k, L_\zeta \to L_\psi) = 2$. The induction hypothesis tells us that $\boldsymbol{m}^\tau(k, \zeta) \leq \boldsymbol{m}^\tau(k, \psi)$, so applying Definition 9 again, we conclude $\boldsymbol{m}^\tau(k, \phi) = 2$. Otherwise, if $\boldsymbol{m}^\tau(k, L_\zeta) > \boldsymbol{m}^\tau(k, L_\psi)$, then by Definition 9, we have $\boldsymbol{m}^\tau(k, L_\zeta \to L_\psi) = \boldsymbol{m}^\tau(k, L_\psi)$. The induction hypothesis then gives us $\boldsymbol{m}^\tau(k, \zeta) > \boldsymbol{m}^\tau(k, \psi)$, so by applying Definition 9, we again obtain $\boldsymbol{m}^\tau(k, \phi) = \boldsymbol{m}^\tau(k, \psi)$.

Downloaded from https://www.cambridge.org/core. UNIVERSIDAD AUTONOMA METROPOLITANA, on 08 Nov 2026 at 14:47:57, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://doi.org/10.1017/S1471068426100714

$\boldsymbol{\phi = \bigcirc_I \psi}$: By equation (24), we have $\mathbf{M} \models \Box(L_\phi \leftrightarrow \bigcirc_I L_\psi)$. From MEL satisfaction, it follows that for any $k \in \lambda$, we have $\mathbf{M}, k \models L_\phi \leftrightarrow \bigcirc_I L_\psi$. By Proposition 17, this gives us $\boldsymbol{m}^\tau(k, L_\phi) = \boldsymbol{m}^\tau(k, \bigcirc_I L_\psi)$.
If $k+1 = \lambda$ or $\tau(k+1) - \tau(k) \notin I$, then by Definition 9, we obtain $\boldsymbol{m}(k, \bigcirc_I L_\psi) = 0$, where also $0 = \boldsymbol{m}^\tau(k, \bigcirc_I \psi)$. Otherwise, we use Definition 9 to get $\boldsymbol{m}(k, \bigcirc_I L_\psi) = \boldsymbol{m}^\tau(k+1, L_\psi)$. Applying the induction hypothesis, we replace $\boldsymbol{m}^\tau(k+1, L_\psi)$ with $\boldsymbol{m}^\tau(k+1, \psi)$, which simplifies to $\boldsymbol{m}^\tau(k, \phi)$ as required.

$\boldsymbol{\phi = \Box_I \psi}$: By equation (24), we obtain $\mathbf{M} \models \Box(L_\phi \leftrightarrow \Box_I L_\psi)$, which implies through MEL satisfaction that $\mathbf{M}, k \models L_\phi \leftrightarrow \Box_I L_\psi$ for all $k \in \lambda$. By Proposition 17, we get $\boldsymbol{m}^\tau(k, L_\phi) = \boldsymbol{m}^\tau(k, \Box_I L_\psi)$. By Definition 9 we know $\boldsymbol{m}^\tau(k, \Box_I L_\psi) = \min\{2, \boldsymbol{m}^\tau(j, L_\psi) \mid k \le j < \lambda \text{ and } \tau(j) - \tau(k) \in I\}$, which by induction can be replaced to get $= \min\{2, \boldsymbol{m}^\tau(j, \psi) \mid k \le j < \lambda \text{ and } \tau(j) - \tau(k) \in I\}$. Finally with Definition 9, this is $\boldsymbol{m}^\tau(k, \Box_I \psi)$.

$\boldsymbol{\phi = \Diamond_I \psi}$: Analogous to previous case. □

*Theorem 5 (Correctness and completeness).*
Let $\varphi$ be a metric formula over $\mathcal{A}$. Then, we have

$$\{(\langle \mathbf{H}, \mathbf{T} \rangle, \tau) \mid (\langle \mathbf{H}, \mathbf{T} \rangle, \tau) \models \varphi\} = \{(\langle \mathbf{H}', \mathbf{T}' \rangle|_{\mathcal{A}}, \tau) \mid (\langle \mathbf{H}', \mathbf{T}' \rangle, \tau) \models \Theta(\varphi)\}$$

*Proof.*
$\subseteq$: Follows from Lemma 9.

$\supseteq$: Take some timed trace $\mathbf{M} = (\langle \mathbf{H}', \mathbf{T}' \rangle, \tau)$ of length $\lambda$ that models $\Theta(\varphi)$, with corresponding metric three-valued interpretation $\boldsymbol{m}^\tau$. Then $\mathbf{M} \models \Box(\mathbf{I} \rightarrow L_\varphi)$ (Definition of satisfaction of MLP) so $\mathbf{M}, 0 \models L_\varphi$ (Satisfaction of always and initial). This implies that $\boldsymbol{m}^\tau(0, L_\varphi) = 2$ (Proposition 15). We can apply Lemma 10 with $\gamma = \varphi$ and $k = 0$ to conclude $\boldsymbol{m}^\tau(0, L_\varphi) = \boldsymbol{m}^\tau(0, \varphi) = 2$ and so, $\mathbf{M}$ is also a model of $\varphi$. Finally, $(\langle \mathbf{H}', \mathbf{T}' \rangle|_{\mathcal{A}}, \tau)$ is still a model of $\varphi$ because the latter is a theory for vocabulary $\mathcal{A}$. □

### *C.2.2 Translation of general metric logic programs to HT*

*Lemma 11 (Splitting general HT).*
Given a metric logic program $P$ and a $\lambda, \nu \in \mathbb{N}$, the stable models of $\boldsymbol{\Pi}_\lambda(P) \cup \Delta_{\lambda,\nu} \cup \boldsymbol{\Psi}_{\lambda,\nu}(P)$ can be computed by means of the splitting technique.

*Proof.*
We will show how the stable models of $\mathbb{P} = \boldsymbol{\Pi}_\lambda(P) \cup \Delta_{\lambda,\nu} \cup \boldsymbol{\Psi}_{\lambda,\nu}(P)$ can be computed by sequentially applying the splitting theorem. Let us start by fixing $U_0 = \{t_{k,d} \mid k, d \in \mathbb{N}\}$. Clearly it is an splitting set because the elements of $U_0$ are only generated by $\Delta_{\lambda,\nu}$. Moreover, $b_{U_0} = \Delta_{\lambda,\nu}$. By Theorem 10, an stable model $T$ of $\mathbb{P}$ can be computed as, $T = T_0 \cup T'$ where $T_0$ is a stable model of $b_{U_0}$. $T'$ is a stable model of $\mathit{Reduce}(\mathbb{P} \setminus b_{U_0}, T_0, U_0 \setminus T_0)$. If we analyze this first simplification, we can assume, without loss of generality, that $\mathit{Reduce}(\mathbb{P} \setminus b_{U_0}, T_0, U_0 \setminus T_0) = \boldsymbol{\Pi}_\lambda(P) \cup \mathit{Reduce}(\boldsymbol{\Psi}_{\lambda,\nu}(P), T_0, U_0 \setminus T_0)$, because $\boldsymbol{\Pi}_\lambda(P)$ does not contain atoms in $U_0$. Let us denote by $\mathbb{P}_1 = \boldsymbol{\Pi}_\lambda(P) \cup \mathit{Reduce}(\boldsymbol{\Psi}_{\lambda,\nu}(P), T_0, U_0 \setminus T_0)$ and let us consider the set $U_1 = \{\delta^{k,j}_{[m..n)} \mid k, j \in \mathbb{N} \text{ and } [m..n) \in \mathbb{I}\,|_{\mathbb{P}}\}$. $U_1$ is also an splitting set because elements of $U_1$ only occur (as facts) in $\mathit{Reduce}(\boldsymbol{\Psi}_{\lambda,\nu}(P), T_0, U_0 \setminus T_0)$. More precisely,

Downloaded from https://www.cambridge.org/core. UNIVERSIDAD AUTONOMA METROPOLITANA, on 08 Nov 2026 at 14:47:57, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://doi.org/10.1017/S1471068426100714

$b_{U_1} = Reduce(\mathbf{\Psi}_{\lambda,\nu}(P), T_0, U_0 \setminus T_0)$. By applying (once again) Theorem 10, we can say that $T' = T_1 \cup T_2$ where $T_1$ is an stable model of $Reduce(\mathbf{\Psi}_{\lambda,\nu}(P), T_0, U_0 \setminus T_0)$ and $T_2$ is a stable model of

$$Reduce(\mathbb{P}_1 \setminus b_{U_1}, T_1, U_1 \setminus T_1) = Reduce(\mathbf{\Pi}_\lambda(P), T_1, U_1 \setminus T_1).$$

□

*Proposition 22 (Equivalences for correspondence).*
Given an HT interpretation $\langle H, T\rangle$ with a corresponding three-valued interpretation $\boldsymbol{m}$, and a timed trace $\mathbf{M} = (\langle \mathbf{H}, \mathbf{T}\rangle, \tau)$ of length $\lambda$ over alphabet $\mathcal{A}$ with a corresponding three-valued metric interpretation $\boldsymbol{m}^\tau$, we have that $\langle H, T\rangle|_{\mathcal{A}^* \cap \mathcal{T}} = \theta(\mathbf{M})$, iff the following properties hold:

1. $\boldsymbol{m}(a_k) = \boldsymbol{m}^\tau(a, k)$ for all $k \in [0..\lambda)$, $a \in \mathcal{A}$
2. $\boldsymbol{m}(t_{k,d}) = 2$ iff $\tau(k) = d$ for all $k \in [0..\lambda)$, $d \in [0..\nu]$
3. $\boldsymbol{m}(t_{k,d}) = 0$ iff $\tau(k) \neq d$ for all $k \in [0..\lambda)$, $d \in [0..\nu]$

where $\nu = \tau(\lambda - 1)$,

*Proof.*
Trivial by construction of $\theta(\mathbf{M})$ □

*Definition 18 (Construction of an HT three-valued interpretation given a metric interpretation).*
Given a timing function $\tau$, a metric three-valued interpretation $\boldsymbol{m}^\tau$, and a program $P$, we construct a three-valued interpretation $\boldsymbol{m}_{[\boldsymbol{m}^\tau, P]}$ over alphabet $\mathcal{A}^* \cup \mathcal{T} \cup \mathcal{L}_P \cup \mathfrak{I}$ as follows:

1. $\boldsymbol{m}_{[\boldsymbol{m}^\tau,P]}(a_k) = \boldsymbol{m}^\tau(a, k)$ for all $a \in \mathcal{A}$
2. $\boldsymbol{m}_{[\boldsymbol{m}^\tau,P]}(t_{k,d}) = 2$ iff $\tau(k) = d$ for all $d \in [0..\nu]$
3. $\boldsymbol{m}_{[\boldsymbol{m}^\tau,P]}(t_{k,d}) = 0$ iff $\tau(k) \neq d$ for all $d \in [0..\nu]$
4. $\boldsymbol{m}_{[\boldsymbol{m}^\tau,P]}(L^k_\mu) = \boldsymbol{m}^\tau(\mu, k)$ for all $\mu \in P$
5. $\boldsymbol{m}_{[\boldsymbol{m}^\tau,P]}(\delta^{k,j}_I) = 2$ if $\tau(j) - \tau(k) \notin I$ for all $I \in \mathbb{I}$
6. $\boldsymbol{m}_{[\boldsymbol{m}^\tau,P]}(\delta^{k,j}_I) = 0$ if $\tau(j) - \tau(k) \in I$ for all $I \in \mathbb{I}$
7. $\boldsymbol{m}_{[\boldsymbol{m}^\tau,P]}(L^{k,j}_{\Box_I b}) = \max\{\boldsymbol{m}_{[\boldsymbol{m}^\tau,P]}(L^j_b), \boldsymbol{m}_{[\boldsymbol{m}^\tau,P]}(\delta^{k,j}_I)\}$ for all $\Box_I b \in P$
8. $\boldsymbol{m}_{[\boldsymbol{m}^\tau,P]}(L^{k,j}_{\Diamond_I b}) = \min\{\boldsymbol{m}_{[\boldsymbol{m}^\tau,P]}(L^j_b), \boldsymbol{m}_{[\boldsymbol{m}^\tau,P]}(\neg\delta^{k,j}_I)\}$ for all $\Diamond_I b \in P$

for all $k \in [0..\lambda)$, $j \in [k..\lambda)$

*Proposition 23 (Construction is total).*
Given a metric three-valued interpretation $\boldsymbol{m}^\tau$, and a program $P$, if $\boldsymbol{m}^\tau$ is total then $\boldsymbol{m}_{[\boldsymbol{m}^\tau,P]}$ is total.

*Proof.*
For $x \in \mathcal{T} \cup \mathfrak{I}$ it is trivial by construction. For $x \in \mathcal{A}^*$ it holds since $\boldsymbol{m}^\tau$ is total. For $x \in \mathcal{L}_P$ it holds recursively since min and max only give values in the set. □

*Definition 19 (Additional interval constraints for monotonic).*

$$\mathbf{\Psi}^\perp_{\lambda,\nu}(P) = \{\perp \leftarrow \delta^{k,j}_I \wedge t_{k,d} \wedge t_{j,d'} \mid 0 \leq k \leq j < \lambda - 1, 0 \leq d < d' \leq \nu, d' - d \in I, I \in \mathcal{I}\} \cup \tag{C4}$$

Downloaded from https://www.cambridge.org/core. UNIVERSIDAD AUTONOMA METROPOLITANA, on 08 Nov 2026 at 14:47:57, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://doi.org/10.1017/S1471068426100714

*Lemma 12 (Modeling of extra constraints).*
Let $P$ be a metric logic program, and $\lambda, \nu \in \mathbb{N}$.

If $\langle T, T\rangle$ is an equilibrium model of $\Delta_{\lambda,\nu} \cup \mathbf{\Psi}_{\lambda,\nu}(P)$, then $\langle T, T\rangle \models \mathbf{\Psi}^{\perp}_{\lambda,\nu}(P)$

*Proof.*
We show that it models each integrity constraint in $\mathbf{\Psi}^{\perp}_{\lambda,\nu}(P)$. Let us assume towards a contradiction that there is constraint $c = \perp \leftarrow \delta^{k,j}_{[m..n)} \wedge t_{k,d} \wedge t_{j,d'}$ that is not modeled by $\langle T, T\rangle$. This implies that $\langle T, T\rangle \models \delta^{k,j}_{[m..n)}$ and $\langle T, T\rangle \models t_{k,d}$ and $\langle T, T\rangle \models t_{j,d'}$, Since $\langle T, T\rangle$ is an equilibrium model of $\Delta_{\lambda,\nu}$, $\langle T, T\rangle$ is timed inducing $\tau$, where $\tau(k) = d$ and $\tau(j) = d'$. Given that $\langle T, T\rangle \models \delta^{k,j}_{[m..n)}$, then it must be founded by a rule in $\mathbf{\Psi}_{\lambda}(P)$, because it is in equilibrium. If it comes from equation (37) then, the condition is that $d' - d < m$, which leads to a contradiction since $d' - d \in I$ is a condition for $c$. If it comes from equation (38) then, the condition is that $d' - d \geq n$, which leads to a contradiction since $d' - d \in I$ is a condition for $c$. □

*Lemma 13 (Completeness lemma HT).*
Let $P$ be a metric logic program over $\mathcal{A}$, and $\mathbf{M} = (\langle \mathbf{H}, \mathbf{T}\rangle, \tau)$ a timed HT-trace of length $\lambda$.

If $\mathbf{M} \models P$, then there exists an HT interpretation $\langle H, T\rangle$ such that $\langle H, T\rangle \models \mathbf{\Pi}_{\lambda}(P) \cup \Delta_{\lambda,\nu} \cup \mathbf{\Psi}_{\lambda,\nu}(P)$ with $\nu = \tau(\lambda - 1)$, and $\langle H, T\rangle|_{\mathcal{A}^* \cup \mathcal{T}} = \theta(\mathbf{M})$.

*Proof.*
We construct $\boldsymbol{m} = \boldsymbol{m}_{[\boldsymbol{m}^{\tau}, P]}$ as done in Definition 18 and let $\langle H, T\rangle$ be the HT interpretation associated to $\boldsymbol{m}$. We notice the properties from Proposition 22 are satisfied by the construction of $\boldsymbol{m}_{[\boldsymbol{m}^{\tau}, P]}$, so we can apply Proposition 22 to assure that $\langle H, T\rangle|_{\mathcal{A}^* \cup \mathcal{T}} = \theta(\mathbf{M})$. We now proceed to show that $\langle H, T\rangle \models \mathbf{\Pi}_{\lambda}(P) \cup \Delta_{\lambda,\nu} \cup \mathbf{\Psi}_{\lambda,\nu}(P)$ by showing that it is a model of each part.

First, we notice that by construction $\langle H, T\rangle$ is timed wrt. $\lambda$ inducing $\tau$. We can apply Proposition 2 to show that $\langle H, T\rangle \models \Delta_{\lambda,\nu}$ for $\nu = \tau(\lambda - 1)$.

Next, we show that $\langle H, T\rangle \models \mathbf{\Psi}_{\lambda,\nu}(P)$ by showing that $\boldsymbol{m}(\mathbf{\Psi}_{\lambda,\nu}(P)) = 2$. Then, we must show that for every rule $r = \delta^{k,j}_{I} \leftarrow t_{k,d} \wedge t_{j,d'}$ in $\mathbf{\Psi}_{\lambda,\nu}(P)$, we have that $\boldsymbol{m}(r) = 2$. Notice that $r$ must come either from equations (37) or (38). The analysis for both cases is the same so we will proceed, with equation (37), and leave equation (38) to the reader. The conditions for equation (37) state that $d' - d < m$, since $\langle H, T\rangle$ it is timed, we have $t_{k,d}$ and $t_{j,d'}$ with $\tau(j) = d'$ and $\tau(k) = d$. So we have that $\tau(j) - \tau(k) \notin I$. If this is the case, then $\boldsymbol{m}(\delta^{k,j}_{I}) = 2$ by construction of $\boldsymbol{m}_{[\boldsymbol{m}^{\tau}, P]}$. Then the three-valued semantics for HT gives us that $\boldsymbol{m}(\delta^{k,j}_{I} \leftarrow t_{k,d} \wedge t_{j,d'}) = 2$, since $2 \geq \{0, 1, 2\}$.

Finally, we show that $\langle H, T\rangle \models \mathbf{\Pi}_{\lambda}(P)$ by showing that $\boldsymbol{m}(\mathbf{\Pi}_{\lambda}(P)) = 2$. Then, we must show that for every rule $\Box r \in P$, and $k \in [0..\lambda)$, we have $\boldsymbol{m}((r)_k) = 2$, and for every $\mu \in P$, we have $\boldsymbol{m}(\eta^*_k(\mu)) = 2$. For the first point, we know by construction of $\boldsymbol{m}_{[\boldsymbol{m}^{\tau}, P]}$, that $\boldsymbol{m}^{\tau}(k, \mu) = \boldsymbol{m}(L^{\mu}_{k})$. With this equivalence we use the substitution to know that if $\boldsymbol{m}^{\tau}(k, r) = 2$ then $\boldsymbol{m}((r)_k) = 2$. And $\boldsymbol{m}^{\tau}(k, r) = 2$ since $\mathbf{M}$ is a model of $P$. For the second point, we can directly apply Proposition 21 given that the requirements are satisfied by Proposition 22. □

Downloaded from https://www.cambridge.org/core. UNIVERSIDAD AUTONOMA METROPOLITANA, on 08 Nov 2026 at 14:47:57, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://doi.org/10.1017/S1471068426100714

*Lemma 14 (Correctness lemma HT).*
Let $P$ be a metric logic program, $\lambda, \nu \in \mathbb{N}$, and $\langle H, T\rangle$ an timed HT interpretation wrt. $\lambda$ inducing $\tau$.

If $\langle H, T\rangle \models \mathbf{\Pi}_\lambda(P) \cup \mathbf{\Psi}_{\lambda,\nu}(P) \cup \mathbf{\Psi}^{\perp}_{\lambda,\nu}(P)$, then $\sigma(\langle H, T\rangle|_{\mathcal{A}^* \cup \mathcal{T}}) \models P$.

*Proof.*
Let $\sigma(\langle H, T\rangle|_{\mathcal{T} \cup \mathcal{A}^*}) = \mathbf{M}$ and let $\boldsymbol{m}$ be the three-valued interpretation corresponding to $\langle H, T\rangle$.

We will show below that each of the conditions of Proposition 21. to prove the pointwise equivalence for the auxiliary atoms, namely $\boldsymbol{m}(L^k_\mu) = \boldsymbol{m}^\tau(k, \mu)$ for all $k \in [0..\lambda)$. With this in hand we argue that $\mathbf{M} \models P$ follows directly from $\langle H, T\rangle \models \mathbf{\Pi}_\lambda(P)$, given that with these equivalence for any $k \in [0..\lambda)$, $\boldsymbol{m}((r)_k) = 2$ iff $\boldsymbol{m}^\tau(k, r) = 2$.

Notice that $\langle H, T\rangle|_{\mathcal{A}^* \cap \mathcal{T}} = \theta(\mathbf{M})$, so we apply Proposition 22 and get $\boldsymbol{m}(a_k) = \boldsymbol{m}^\tau(k, a)$ for all $k \in [0..\lambda)$, $a \in \mathcal{A}$.

We will now show that $\boldsymbol{m}(\delta^{k,j}_I) = 2$ iff $\tau(j) - \tau(k) \notin I$ for all $k \in [0..\lambda)$, $I \in \mathbb{I}$. For this let $k \in [0..\lambda)$ and $I = [m..n) \in \mathbb{I}$. We know that $\langle H, T\rangle \models t_{k,d}$ and $\langle H, T\rangle \models t_{j,d'}$ with $d' > d$, where $\tau(k) = d$ and $\tau(j) = d'$, since $\langle H, T\rangle$ is timed wrt. $\lambda$. This shows that $\boldsymbol{m}(t_{k,d} \wedge t_{j,d'}) = 2$, since $\boldsymbol{m}(t_{k,d}) = 2$ and $\boldsymbol{m}(t_{j,d'}) = 2$. Now we prove both directions of the equivalence:

$\leftarrow$: We start with $\tau(j) - \tau(k) \notin I$, which gives us the conditions in equations (37) or (38) depending on the two cases for not being in $I$. Then, $\boldsymbol{m}(\delta^{k,j}_I \leftarrow t_{k,d} \wedge t_{j,d'}) = 2$ since $\langle H, T\rangle \models \mathbf{\Psi}_{\lambda,\nu}(P)$. By using the three-valued semantics for HT we know that $\boldsymbol{m}(\delta^{k,j}_I) = 2$, since $\boldsymbol{m}(t_{k,d} \wedge t_{j,d'}) = 2$.

$\rightarrow$: We start with $\boldsymbol{m}(\delta^{k,j}_I) = 2$, and let's assume towards a contradiction that $\tau(j) - \tau(k) \in I$. Then, $\boldsymbol{m}(\bot \leftarrow \delta^{k,j}_{[m..n)} \wedge t_{k,d} \wedge t_{j,d'}) = 2$ since $\langle H, T\rangle \models \mathbf{\Psi}^{\perp}_{\lambda,\nu}(P)$. With three-valued semantics for HT we know that $\boldsymbol{m}(\delta^{k,j}_{[m..n)} \wedge t_{k,d} \wedge t_{j,d'}) = 0$, which leads to a contradiction, since $\boldsymbol{m}(\delta^{k,j}_I) = 2$ and $\boldsymbol{m}(t_{k,d} \wedge t_{j,d'}) = 2$.

Therefore, $\boldsymbol{m}(\delta^{k,j}_I) = 2$ iff $\tau(j) - \tau(k) \notin I$ for all $k \in [0..\lambda)$, $I \in \mathbb{I}$

We will now show that $\boldsymbol{m}(\delta^{k,j}_I) = 0$ iff $\tau(j) - \tau(k) \in I$ for all $k \in [0..\lambda)$, $I \in \mathbb{I}$. For this let $k \in [0..\lambda)$ and $I = [m..n) \in \mathbb{I}$. As before, we know that $\langle H, T\rangle \models t_{k,d}$ and $\langle H, T\rangle \models t_{j,d'}$, with $d' > d$, where $\tau(k) = d$ and $\tau(j) = d'$, since it is timed. Now we prove both directions of the equivalence:

$\leftarrow$: We start with $\tau(j) - \tau(k) \in I$. Since $\langle H, T\rangle \models \mathbf{\Psi}^{\perp}_{\lambda,\nu}(P)$, then $\boldsymbol{m}(\bot \leftarrow \delta^{k,j}_{[m..n)} \wedge t_{k,d} \wedge t_{j,d'}) = 2$, which means that $\boldsymbol{m}(\delta^{k,j}_{[m..n)} \wedge t_{k,d} \wedge t_{j,d'}) = 0$. The three-valued semantics for HT gives us that $\min\{\boldsymbol{m}(\delta^{k,j}_{[m..n)}), 2\} = 0$, after substituting the known values. Then $\boldsymbol{m}(\delta^{k,j}_I) = 0$.

$\rightarrow$: We start with $\boldsymbol{m}(\delta^{k,j}_I) = 0$, and assume towards a contradiction that $\tau(j) - \tau(k) \notin I$. Then $\boldsymbol{m}(\delta^{k,j}_I \leftarrow t_{k,d} \wedge t_{j,d'}) = 2$ since $\langle H, T\rangle \models \mathbf{\Psi}_{\lambda,\nu}(P)$ and we can apply equations (37) or (38) depending on the two cases for not being in $I$. Contradiction since $\boldsymbol{m}(t_{k,d} \wedge t_{j,d'}) = 2$.

Therefore, $\boldsymbol{m}(\delta^{k,j}_I) = 0$ iff $\tau(j) - \tau(k) \in I$ for all $k \in [0..\lambda)$, $I \in \mathbb{I}$.

Downloaded from https://www.cambridge.org/core. UNIVERSIDAD AUTONOMA METROPOLITANA, on 08 Nov 2026 at 14:47:57, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://doi.org/10.1017/S1471068426100714

We will now show that that $\boldsymbol{m}(L_\mu^{k,j})=\max\{\boldsymbol{m}(L_b^j),\boldsymbol{m}(\delta_I^{k,j})\}$. Consider $\mu=\Box_I b$ and let $k\in[0..\lambda)$, $j\in[k..\lambda)$. Since $\langle h,t\rangle\models\mathbf{\Pi}_\lambda(P)$, we have that $\boldsymbol{m}(\eta_k(\mu))=2$. We use the definition of Table 3 and get $\boldsymbol{m}(L_\mu^{k,j}\leftrightarrow L_b^j\vee\delta_I^{k,j})=2$. By applying Proposition 8, $\boldsymbol{m}(L_\mu^{k,j})=\boldsymbol{m}(L_b^j\vee\delta_I^{k,j})$, which by the semantics of three-valued logic gives us that $\boldsymbol{m}(L_\mu^{k,j})=\max\{\boldsymbol{m}(L_b^j),\boldsymbol{m}(\delta_I^{k,j})\}$. The prove that $\boldsymbol{m}(L_\mu^{k,j})=\min\{\boldsymbol{m}(L_b^j),\boldsymbol{m}(\neg\delta_I^{k,j})\}$, follows analogously. □

*Theorem 6 (Completeness).*
Let $P$ be a metric logic program over $\mathcal{A}$, and $\mathbf{M}=(\langle\mathbf{T},\mathbf{T}\rangle,\tau)$ a total timed HT-trace of length $\lambda$.

If $\mathbf{M}$ is a metric equilibrium model of $P$, then there exists an equilibrium model $\langle T,T\rangle$ of $\mathbf{\Pi}_\lambda(P)\cup\Delta_{\lambda,\nu}\cup\mathbf{\Psi}_{\lambda,\nu}(P)$ with $\nu=\tau(\lambda-1)$ such that $\langle T,T\rangle|_{\mathcal{A}^*\cap\mathcal{T}}=\theta(\mathbf{M})$.

*Proof.*
We construct $\langle T,T\rangle$ such that its corresponding three-valued interpretation is $\boldsymbol{m}_{[\boldsymbol{m}^\tau,P]}$. Notice the properties from Proposition 22 are satisfied by the construction of $\boldsymbol{m}_{[\boldsymbol{m}^\tau,P]}$, so we can apply Proposition 22 to assure that $\langle T,T\rangle|_{\mathcal{A}^*\cup\mathcal{T}}=\theta(\mathbf{M})$. In what follows, we will prove the theorem by proving incrementally that $\langle T,T\rangle$ is an equilibrium model of each part of the program using Lemma 11, limiting the alphabet to the corresponding atoms.

We begin proving that $\langle T,T\rangle|_{\mathcal{T}}$ is an equilibrium model of $\Delta_{\lambda,\nu}$ Notice that by construction $\langle T,T\rangle$ is timed wrt. $\lambda$ inducing $\tau$. Therefore, $\langle T,T\rangle|_{\mathcal{T}}$ is an equilibrium model of $\Delta_{\lambda,\nu}$ by Proposition 2.

We will now prove that $\langle T,T\rangle|_{\mathcal{T}\cup\mathfrak{I}}$ is an equilibrium model of $\Delta_{\lambda,\nu}\cup\mathbf{\Psi}_{\lambda,\nu}(P)$. We can prove that $\langle T,T\rangle|_{\mathcal{T}\cup\mathfrak{I}}\models\mathbf{\Psi}_{\lambda,\nu}(P)$ as in the proof of Lemma 12. For the equilibrium criteria, we assume towards a contradiction that there is $\langle H,T\rangle$ such that $H\subset T$ and $\langle H,T\rangle\models\mathbf{\Psi}_{\lambda,\nu}(P)$, and let $\boldsymbol{m}'$ be its corresponding three-valued interpretation. This means that there is $x\in\mathfrak{I}$ such that $x\in T$ and $x\notin H$, where $\boldsymbol{m}'(x)=1$. Then, $x=\delta_I^{k,j}$ for some $k\in[0..\lambda)$, $j\in[k..\lambda)$ and $I\in\mathbb{I}$. Since $x\in T$, by construction of $\boldsymbol{m}_{[\boldsymbol{m}^\tau,P]}$, $\tau(j)-\tau(k)\notin I$, we get that $\boldsymbol{m}_{[\boldsymbol{m}^\tau,P]}(\delta_I^{k,j})=2$. Since it is timed wrt. $\lambda$, then $d'-d\notin I$ and $\boldsymbol{m}'(t_{k,d}\wedge t_{j,d'})=2$. We also know that $\boldsymbol{m}'(\delta_{[m..n)}^{k,j}\leftarrow t_{k,d}\wedge t_{j,d'})=2$ since $\langle H,T\rangle\models\mathbf{\Psi}_{\lambda,\nu}(P)$. Then, by the three-valued semantics for HT we know that $\boldsymbol{m}'(\delta_{[m..n)}^{k,j})=2$. This leads to a contradiction since $\boldsymbol{m}'(x)=1$. Therefore, $\langle T,T\rangle|_{\mathcal{T}\cup\mathfrak{I}}$ is an equilibrium model of $\Delta_{\lambda,\nu}\cup\mathbf{\Psi}_{\lambda,\nu}(P)$.

We will now prove that $\langle T,T\rangle$ is an equilibrium model of $\mathbf{\Pi}_\lambda(P)\cup\Delta_{\lambda,\nu}\cup\mathbf{\Psi}_{\lambda,\nu}(P)$. We can proof that $\langle T,T\rangle\models\mathbf{\Pi}_\lambda(P)$ as in the proof of Lemma 13. We assume towards a contradiction that there is $\langle H,T\rangle$ such that $H\subset T$ and $\langle H,T\rangle\models\mathbf{\Pi}_\lambda(P)$, and let $\boldsymbol{m}'$ be its corresponding three-valued interpretation. Then, there is $x\in\mathcal{A}^*\cup\mathcal{L}_P$ such that $x\in T$ and $x\notin H$ so $\boldsymbol{m}'(x)=1$. We analyze the cases for $x$:

$\boldsymbol{x\in\mathcal{A}^*}$: $x=a_k$ for some $0\le k<\lambda$. since $\langle T,T\rangle|_{\mathcal{T}\cup\mathfrak{I}}$ is an equilibrium model of $\Delta_{\lambda,\nu}\cup\mathbf{\Psi}_{\lambda,\nu}(P)$, we can apply Lemma 12 to get that $tandt|_{\mathcal{T}\cup\mathfrak{I}}\models\mathbf{\Psi}^\perp_{\lambda,\nu}(P)$, and since $\langle H,T\rangle|_{\mathcal{T}\cup\mathfrak{I}}=\langle T,T\rangle|_{\mathcal{T}\cup\mathfrak{I}}$, we get that $\langle H,T\rangle\models\mathbf{\Psi}^\perp_{\lambda,\nu}(P)$. We also know that $\langle H,T\rangle\models\mathbf{\Pi}_\lambda(P)\cup\mathbf{\Psi}_{\lambda,\nu}(P)$, so by Lemma 14, we get that $\sigma(\langle H,T\rangle|_{\mathcal{A}^*\cup\mathcal{T}})\models P$. This leads to a contradiction since $\mathbf{M}$ is an equilibrium model of $P$ but $\sigma(\langle H,T\rangle|_{\mathcal{A}^*\cup\mathcal{T}})<\mathbf{M}$. Therefore, $\langle T,T\rangle|_{\mathcal{A}^*}=\langle H,T\rangle|_{\mathcal{A}^*}$

Downloaded from https://www.cambridge.org/core. UNIVERSIDAD AUTONOMA METROPOLITANA, on 08 Nov 2026 at 14:47:57, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://doi.org/10.1017/S1471068426100714

$\boldsymbol{x \in \mathcal{L}_P}$: Notice that if $\langle H, T\rangle \models \mathbf{\Pi}_\lambda(P)$ then the value of $x \in \mathcal{L}_P$ is fully defined by the atoms in $\mathcal{A}^* \cup \mathfrak{I}$. Since $\langle T, T\rangle|_{\mathcal{A}^*\cup\mathfrak{I}} = \langle H, T\rangle|_{\mathcal{A}^*\cup\mathfrak{I}}$, then, $\boldsymbol{m}'(x) = \boldsymbol{m}_{[\boldsymbol{m}^\tau, P]}(x)$. But since $\langle T, T\rangle$ is total then $\boldsymbol{m}_{[\boldsymbol{m}^\tau, P]}(x)$ is either 0 or 1, but $\boldsymbol{m}'(x) = 1$. This leads to a contradiction.

Therefore, $\langle T, T\rangle$ is an equilibrium model of $\mathbf{\Pi}_\lambda(P) \cup \Delta_{\lambda,\nu} \cup \mathbf{\Psi}_{\lambda,\nu}(P)$. □

*Theorem 7 (Correctness).*
Let $P$ be a metric logic program, and $\lambda, \nu \in \mathbb{N}$.

If $\langle T, T\rangle$ is an equilibrium model of $\mathbf{\Pi}_\lambda(P) \cup \Delta_{\lambda,\nu} \cup \mathbf{\Psi}_{\lambda,\nu}(P)$, then $\sigma(\langle T, T\rangle|_{\mathcal{A}^*\cap\mathcal{T}})$ is a metric equilibrium model of $P$.

*Proof.*
Let $\sigma(\langle T, T\rangle|_{\mathcal{T}\cup\mathcal{A}^*}) = \mathbf{M}$.

We begin proving that $\mathbf{M} \models P$. We do so by using Lemma 14, so it remains proving that the conditions for this Lemma hold. Firstly, $\langle T, T\rangle|_{\mathcal{T}}$ is an equilibrium model of $\Delta_\lambda$ given the splitting property in Lemma 11. Due to Proposition 1, $\langle T, T\rangle|_{\mathcal{T}}$ is timed wrt. $\lambda$ inducing $\tau$. Since adding other atoms not in $\mathcal{T}$ keeps timed property, $\langle T, T\rangle$ is also timed. For the second condition, $\langle T, T\rangle|_{\mathcal{T}\cup\mathfrak{I}}$ is an equilibrium model of $\Delta_\lambda \cup \mathbf{\Psi}_\lambda(P)$ thanks to the Splitting property. Applying Lemma 12 we get that $\langle T, T\rangle|_{\mathcal{T}\cup\mathfrak{I}} \models \mathbf{\Psi}^\perp_\lambda(P)$. As before, we can add the rest of the atoms since there is no interference so $\langle T, T\rangle \models \mathbf{\Psi}^\perp_\lambda(P)$. We also know that $\langle T, T\rangle \models \mathbf{\Pi}_\lambda(P) \cup \mathbf{\Psi}_{\lambda,\nu}(P)$ from the hypothesis. With this we can apply Lemma 14 to get that $\mathbf{M} \models P$.

Now we prove that it is in equilibrium. By construction $\mathbf{M}$ is total. We assume towards a contradiction that there is $\mathbf{M}' = (\langle \mathbf{H}, \mathbf{T}\rangle, \tau)$ such that $\mathbf{H} < \mathbf{T}$ and $\mathbf{M}' \models P$. Then, there is $a \in \mathcal{A}$ such that $a \in T_k$ and $a \notin H_k$. We apply Lemma 13, and let $\langle H, T\rangle$ be an HT interpretation such that $\langle H, T\rangle \models \mathbf{\Pi}_\lambda(P) \cup \Delta_{\lambda,\nu} \cup \mathbf{\Psi}_{\lambda,\nu}(P)$ with $\nu = \tau(\lambda - 1)$, and $\langle H, T\rangle|_{\mathcal{A}^*\cup\mathcal{T}} = \theta(\mathbf{M}')$ By construction $a_k \in T$ and $a_k \notin H$, which means that $\langle H, T\rangle < \langle T, T\rangle$. Leading to a contradiction since $\langle T, T\rangle$ is an equilibrium model. □

### *C.2.3 Translation of general metric logic programs to HT$_c$*

*Lemma 15 (Splitting general HT$_c$).*
Given a metric logic program $P$ and a $\lambda \in \mathbb{N}$, the stable models of $\mathbf{\Pi}_\lambda(P) \cup \Delta^c_\lambda \cup \mathbf{\Psi}^c_\lambda(P)$ can be computed by means of the splitting technique.

*Proof.*
Proven as Lemma 11 but using the notion of splitting set as defined in Cabalar *et al.* (2020c, Definition 10). □

*Proposition 24 (Equivalences for correspondence).*
Given an HT$_c$ interpretation $\langle h, t\rangle$, a timed trace $\mathbf{M} = (\langle \mathbf{H}, \mathbf{T}\rangle, \tau)$ of length $\lambda$ over alphabet $\mathcal{A}$, and its corresponding three-valued metric interpretation $\boldsymbol{m}^\tau$, we have that $\langle h, t\rangle|_{\mathcal{A}^*\cap\mathcal{T}^c} = \theta^c(\mathbf{M})$, iff the following properties hold:

1. $\boldsymbol{m}^{\langle h,t\rangle}(a_k = \boldsymbol{t}) = \boldsymbol{m}^\tau(a, k)$ for all $k \in [0..\lambda)$, $a \in \mathcal{A}$
2. $\boldsymbol{m}^{\langle h,t\rangle}(t_k = d) = 2$ iff $\tau(k) = d$ for all $k \in [0..\lambda)$, $d \in \mathbb{N}$
3. $\boldsymbol{m}^{\langle h,t\rangle}(t_k = d) = 0$ iff $\tau(k) \neq d$ for all $k \in [0..\lambda)$, $d \in \mathbb{N}$

Downloaded from https://www.cambridge.org/core. UNIVERSIDAD AUTONOMA METROPOLITANA, on 08 Nov 2026 at 14:47:57, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://doi.org/10.1017/S1471068426100714

*Proof.*
Trivial by construction of $\theta^c(\mathbf{M})$ □

*Definition 20 (Construction of an HTc three-valued interpretation given a metric interpretation).*
Given a timing function $\tau$, a metric three-valued interpretation $\boldsymbol{m}^\tau$, and a program $P$, construct an $\mathrm{HT}_c$ interpretation, $\langle h, t\rangle = \chi(\boldsymbol{m}^\tau, P)$ over alphabet $\mathcal{A}^* \cup \mathcal{T}^c \cup \mathcal{L}_P \cup \mathfrak{I}$ as follows:

$$h(t_k) = t(t_k) = \tau(k)$$

$$h(\delta_I^{k,j}) = t(\delta_I^{k,j}) = \begin{cases} \boldsymbol{t} & \text{if } \tau(j) - \tau(k) \notin I \\ \boldsymbol{u} & \text{otherwise} \end{cases} \quad \text{for all interval } I \in \mathbb{I}$$

$$h(a_k) = \begin{cases} \boldsymbol{t} & \text{if } \boldsymbol{m}^\tau(k, a) = 2 \\ \boldsymbol{u} & \text{otherwise} \end{cases} \quad \text{for all atom } a \in \mathcal{A}$$

$$t(a_k) = \begin{cases} \boldsymbol{t} & \text{if } \boldsymbol{m}^\tau(k, a) > 0 \\ \boldsymbol{u} & \text{otherwise} \end{cases} \quad \text{for all atom } a \in \mathcal{A}$$

$$h(L_\mu^k) = \begin{cases} \boldsymbol{t} & \text{if } \boldsymbol{m}^\tau(k, \mu) = 2 \\ \boldsymbol{u} & \text{otherwise} \end{cases} \quad \text{for all } \mu \in P$$

$$t(L_\mu^k) = \begin{cases} \boldsymbol{t} & \text{if } \boldsymbol{m}^\tau(k, \mu) > 0 \\ \boldsymbol{u} & \text{otherwise} \end{cases} \quad \text{for all } \mu \in P$$

$$h'(L_{\Box_I b}^{k,j}) = \begin{cases} \boldsymbol{t} & \text{if } h'(L_b^j) = \boldsymbol{t} \text{ or } h'(\delta_I^{k,j}) = \boldsymbol{t} \\ \boldsymbol{u} & \text{otherwise} \end{cases} \quad \text{for all } \Box_I b \in P$$

$$h'(L_{\Diamond_I b}^{k,j}) = \begin{cases} \boldsymbol{t} & \text{if } h'(L_b^j) = \boldsymbol{t} \text{ and } h'(\delta_I^{k,j}) = \boldsymbol{u} \\ \boldsymbol{u} & \text{otherwise} \end{cases} \quad \text{for all } \Diamond_I b \in P$$

for all $k \in [0..\lambda)$, $j \in [k..\lambda)$, $h' \in \{h, t\}$

*Proposition 25 (Properties of the three-valued interpretation).*
Given a timing function $\tau$, a metric three-valued interpretation $\boldsymbol{m}^\tau$, a program $P$, and its corresponding $\mathrm{HT}_c$ interpretation $\chi(\boldsymbol{m}^\tau, P)$, then $\boldsymbol{m}^{\chi(\boldsymbol{m}^\tau, P)}$ An HT-interpretation

1. $\boldsymbol{m}^{\chi(\boldsymbol{m}^\tau, P)}(a_k = \boldsymbol{t}) = \boldsymbol{m}^\tau(k, a)$ for all $a \in \mathcal{A}$
2. $\boldsymbol{m}^{\chi(\boldsymbol{m}^\tau, P)}(t_k = d) = 2$ iff $\tau(k) = d$ for all $d \in \mathbb{N}$
3. $\boldsymbol{m}^{\chi(\boldsymbol{m}^\tau, P)}(t_k = d) = 0$ iff $\tau(k) \neq d$ for all $d \in \mathbb{N}$
4. $\boldsymbol{m}^{\chi(\boldsymbol{m}^\tau, P)}(L_\mu^k = \boldsymbol{t}) = \boldsymbol{m}^\tau(k, \mu)$ for all $\mu \in P$
5. $\boldsymbol{m}^{\chi(\boldsymbol{m}^\tau, P)}(\delta_I^{k,j} = \boldsymbol{t}) = 2$ iff $\tau(j) - \tau(k) \notin I$ for all $I \in \mathbb{I}$
6. $\boldsymbol{m}^{\chi(\boldsymbol{m}^\tau, P)}(\delta_I^{k,j} = \boldsymbol{t}) = 0$ iff $\tau(j) - \tau(k) \in I$ for all $I \in \mathbb{I}$
7. $\boldsymbol{m}^{\chi(\boldsymbol{m}^\tau, P)}(L_{\Box_I b}^{k,j} = \boldsymbol{t}) = \max\{\boldsymbol{m}^{\chi(\boldsymbol{m}^\tau, P)}(L_b^j = \boldsymbol{t}), \boldsymbol{m}^{\chi(\boldsymbol{m}^\tau, P)}(\delta_I^{k,j} = \boldsymbol{t})\}$ for all $\Box_I b \in P$
8. $\boldsymbol{m}^{\chi(\boldsymbol{m}^\tau, P)}(L_{\Diamond_I b}^{k,j} = \boldsymbol{t}) = \min\{\boldsymbol{m}^{\chi(\boldsymbol{m}^\tau, P)}(L_b^j = \boldsymbol{t}), \boldsymbol{m}^{\chi(\boldsymbol{m}^\tau, P)}(\neg(\delta_I^{k,j} = \boldsymbol{t}))\}$ for all $\Diamond_I b \in P$

for all $k \in [0..\lambda)$, $j \in [k..\lambda)$

Downloaded from https://www.cambridge.org/core. UNIVERSIDAD AUTONOMA METROPOLITANA, on 08 Nov 2026 at 14:47:57, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://doi.org/10.1017/S1471068426100714

*Proof.*
Trivial by construction of $\chi(\boldsymbol{m}^\tau, P)$ in Definition 20. □

*Proposition 26 (Construction is total).*
Given a metric three-valued interpretation $\boldsymbol{m}^\tau$, and a program $P$, if $\boldsymbol{m}^\tau$ is total then $\boldsymbol{m}^{\chi(\boldsymbol{m}^\tau, P)}$ is total.

*Proof.*
For $x \in \mathcal{T}^c \cup \mathfrak{I}$ it is trivial by construction. For $x \in \mathcal{A}^*$ it holds since $\boldsymbol{m}^\tau$ is total. For $x \in \mathcal{L}_P$ it holds recursively since min and max only give values in the set. □

*Definition 21 (Additional interval constraints for monotonic).*

$$\boldsymbol{\Psi}_\lambda^{c\perp}(P) = \{\bot \leftarrow \delta^{k,j}_{[m..n)} \wedge (t_k - t_j \leq -m) \wedge (t_j - t_k \leq n-1) \mid 0 \leq k \leq j < \lambda - 1, [m..n) \in \mathcal{I}\} \cup \tag{C5}$$

$$\{\bot \leftarrow \delta^{k,j}_{[m..\omega)} \wedge (t_k - t_j \leq -m) \mid 0 \leq k \leq j < \lambda - 1, [m..\omega) \in \mathcal{I}\} \tag{C6}$$

*Lemma 16 (Modeling of extra constraints).*
Let $P$ be a metric logic program, and $\lambda \in \mathbb{N}$.
If $\langle t, t\rangle$ is an constraint equilibrium model of $\Delta_\lambda^c \cup \boldsymbol{\Psi}_\lambda^c(P)$, then $\langle t, t\rangle \models \boldsymbol{\Psi}_\lambda^{\perp,c}(P)$

*Proof.*
We show that it models each rule in $\boldsymbol{\Psi}_\lambda^{c\perp}(P)$.

Let us assume towards a contradiction that there is a rule $c \in \boldsymbol{\Psi}_\lambda^{c\perp}(P)$ such that $\langle t, t\rangle \not\models c$. Let $c = \bot \leftarrow \delta^{k,j}_{[m..n)} \wedge (t_k - t_j \leq -m) \wedge (t_j - t_k \leq n-1)$.[15] Then $\langle t, t\rangle \models \delta^{k,j}_{[m..n)}$ and $\langle t, t\rangle \models t_k - t_j \leq -m$ and $\langle t, t\rangle \models t_j - t_k \leq n-1$.

Since $\langle t, t\rangle$ is an equilibrium model of $\Delta_\lambda^c$, $\langle t, t\rangle \models \Delta_\lambda^c$. By applying Proposition 4, we get that $\langle t, t\rangle$ is timed inducing $\tau$, and $\tau(k) = t(t_k)$ and $\tau(j) = t(t_j)$. Given that $\langle t, t\rangle \models \delta^{k,j}_{[m..n)}$, then it must be founded by a rule in $\boldsymbol{\Psi}_\lambda^c(P)$, because it is in equilibrium. If it comes from equation (41) then $\langle t, t\rangle \models \neg(t_k - t_j \leq -m)$ and $\langle t, t\rangle \not\models t_k - t_j \leq -m$, which leads to a contradiction. If it comes from equation (42) then $\langle t, t\rangle \models \neg(t_j - t_k \leq n-1)$ and $\langle t, t\rangle \not\models t_j - t_k \leq n-1$, which leads to a contradiction as well. □

*Lemma 17 (Completeness lemma HTC).*
Let $P$ be a metric logic program over $\mathcal{A}$, and $\mathbf{M} = (\langle \mathbf{H}, \mathbf{T}\rangle, \tau)$ a timed HT-trace of length $\lambda$.

If $\mathbf{M} \models P$, then there exists an $\mathrm{HT}_c$ interpretation $\langle h, t\rangle$ such that $\langle h, t\rangle \models \boldsymbol{\Pi}_\lambda(P) \cup \Delta_\lambda^c \cup \boldsymbol{\Psi}_\lambda^c(P)$ and $\langle h, t\rangle|_{\mathcal{A}^* \cup \mathcal{T}^c} = \theta^c(\mathbf{M})$.

*Proof.*
Let $\boldsymbol{m}^\tau$ be the three-valued metric interpretation corresponding to $\mathbf{M}$ as in Definition 10. We construct $\langle h, t\rangle = \chi(\boldsymbol{m}^\tau, P)$ following Definition 20. We notice the properties from Proposition 24 are satisfied by the construction of $\boldsymbol{m}^{\langle h,t\rangle}$ (Proposition 25), so we can apply Proposition 24 to assure that $\langle h, t\rangle|_{\mathcal{A}^* \cup \mathcal{T}} = \theta^c(\mathbf{M})$. We now proceed to show that $\langle h, t\rangle \models \boldsymbol{\Pi}_\lambda(P) \cup \Delta_\lambda^c \cup \boldsymbol{\Psi}_\lambda^c(P)$ by showing that it is a model of each part.

[15] Notice that the case for $[m..\omega)$ is a simpler version of this case as it has less conditions, so will not be considered here.

Downloaded from https://www.cambridge.org/core. UNIVERSIDAD AUTONOMA METROPOLITANA, on 08 Nov 2026 at 14:47:57, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://doi.org/10.1017/S1471068426100714

Notice that by construction $\langle h, t\rangle$ is timed wrt. $\lambda$ inducing $\tau$. We can apply Proposition 5 to show that $\langle h, t\rangle \models \Delta^c_\lambda$.

Next, we show that $\langle h, t\rangle \models \mathbf{\Psi}^c_\lambda(P)$ by showing that $\boldsymbol{m}^{\langle h,t\rangle}(\mathbf{\Psi}^c_\lambda(P)) = 2$. Then, we must show that for every rule $r \in \mathbf{\Psi}^c_\lambda(P)$, we have that $\boldsymbol{m}^{\langle h,t\rangle}(r) = 2$. Notice that $h(t_k) = \tau(k)$ and $h(t_j) = \tau(j)$, since it is timed. We proof this by case analysis on $r$

$\boldsymbol{\delta^{k,j}_{[m..n)} \leftarrow \neg(t_k - t_j \leq -m)}$: Now we analyze the two cases below.

$\boldsymbol{\tau(k) - \tau(j) > -m}$: We then know that $\tau(k) - \tau(j) \notin [m..n)$. By Construction and Proposition 25 then $\boldsymbol{m}^{\langle h,t\rangle}(\delta^{k,j}_I) = 2$. Since $2 \geq \max\{0, 1, 2\}$, then $\boldsymbol{m}^{\langle h,t\rangle}(\delta^{k,j}_I) \geq \boldsymbol{m}^{\langle h,t\rangle}(\neg(t_k - t_j \leq -m))$. Finally, the three-valued semantics for $\mathrm{HT}_c$ gives us that $\boldsymbol{m}^{\langle h,t\rangle}(\delta^{k,j}_I \leftarrow \neg(t_k - t_j \leq -m)) = 2$.

$\boldsymbol{\tau(k) - \tau(j) \leq -m}$: We then know that $\boldsymbol{m}^{\langle h,t\rangle}(t_k - t_j \leq -m) = 2$. The three-valued semantics for $\mathrm{HT}_c$ gives us that $\boldsymbol{m}^{\langle h,t\rangle}(\neg t_k - t_j \leq -m) = 0$, and consequently $\boldsymbol{m}^{\langle h,t\rangle}(\delta^{k,j}_I \leftarrow \neg(t_k - t_j \leq -m)) = 2$.

$\boldsymbol{\delta^{k,j}_{[m..n)} \leftarrow \neg(t_j - t_k \leq n - 1)}$: Analogous

Finally, we show that $\langle h, t\rangle \models \mathbf{\Pi}_\lambda(P)$ by showing that $\boldsymbol{m}^{\langle h,t\rangle}(\mathbf{\Pi}_\lambda(P)) = 2$. Then, we must show that for every rule $\Box r \in P$, and $k \in [0..\lambda)$, we have $\boldsymbol{m}^{\langle h,t\rangle}((r)_k) = 2$, and for every $\mu \in P$, we have $\boldsymbol{m}^{\langle h,t\rangle}(\eta^*_k(\mu)) = 2$. For the first point, we know by construction of $\chi(\boldsymbol{m}^\tau, P)$ and Proposition 25, that $\boldsymbol{m}^\tau(k, \mu) = \boldsymbol{m}^{\langle h,t\rangle}(L^\mu_k)$. With this equivalence know that if $\boldsymbol{m}^\tau(k, r) = 2$ then $\boldsymbol{m}^{\langle h,t\rangle}((r)_k) = 2$. And $\boldsymbol{m}^\tau(k, r) = 2$ since $\mathbf{M}$ is a model of $P$. For the second point, again by construction, $\boldsymbol{m}^{\langle h,t\rangle}(L^k_\mu = \boldsymbol{t}) = \boldsymbol{m}^\tau(k, \mu)$. We can then apply Proposition 21, given that the requirements are satisfied by Proposition 25.[16] With this we get $\boldsymbol{m}^{\langle h,t\rangle}(\eta^*_k(\mu) = \boldsymbol{t}) = 2$. □

*Lemma 18 (Correctness lemma HT).*
Let $P$ be a metric logic program, $\lambda \in \mathbb{N}$, and $\langle h, t\rangle$ an timed $\mathrm{HT}_c$ interpretation wrt. $\lambda$
If $\langle h, t\rangle \models \mathbf{\Pi}_\lambda(P) \cup \mathbf{\Psi}^c_\lambda(P) \cup \mathbf{\Psi}^{c\perp}_\lambda(P)$, then $\sigma^c(\langle h, t\rangle|_{\mathcal{A}^* \cup \mathcal{T}^c}) \models P$.

*Proof.*
Let $\sigma^c(\langle h, t\rangle|_{\mathcal{T}^c \cup \mathcal{A}^*}) = \mathbf{M}$, and let $\tau$ be the timing function induced by $\langle h, t\rangle$.

We will show below that each of the conditions of Proposition 21 hold, to prove the point-wise equivalence for the auxiliary atoms, namely $\boldsymbol{m}^{\langle h,t\rangle}(L^k_\mu) = \boldsymbol{m}^\tau(k, \mu)$ for all $k \in [0..\lambda)$. With this in hand we argue that $\mathbf{M} \models P$ follows directly from $\langle h, t\rangle \models \mathbf{\Pi}_\lambda(P)$, given that with these equivalence for any $k \in [0..\lambda)$, $\boldsymbol{m}^{\langle h,t\rangle}((r)_k) = 2$ iff $\boldsymbol{m}^\tau(k, r) = 2$.

Notice $\langle h, t\rangle|_{\mathcal{A}^* \cap \mathcal{T}^c} = \theta^c(\mathbf{M})$, so we apply Proposition 24 and get $\boldsymbol{m}^{\langle h,t\rangle}(a_k = \boldsymbol{t}) = \boldsymbol{m}^\tau(k, a)$ for all $k \in [0..\lambda)$, $a \in \mathcal{A}$.

We will now show that $\boldsymbol{m}^{\langle h,t\rangle}(\delta^{k,j}_I = \boldsymbol{t}) = 2$ iff $\tau(j) - \tau(k) \notin I$ for all $k \in [0..\lambda)$, $I \in \mathbb{I}$. For this let $k \in [0..\lambda)$ and $I = [m..n) \in \mathbb{I}$. We know that $h(t_k) = \tau(k)$ and $h(t_j) = \tau(j)$, since $\langle h, t\rangle$ is timed inducing $\tau$. Now we prove both directions of the equivalence:

[16] Notice that Proposition 21 is defined for HT but it holds the same for $\mathrm{HT}_c$ since it uses only boolean variables. (Observation 1 in Cabalar *et al.* (2016)).

Downloaded from https://www.cambridge.org/core. UNIVERSIDAD AUTONOMA METROPOLITANA, on 08 Nov 2026 at 14:47:57, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://doi.org/10.1017/S1471068426100714

$\leftarrow$: We start with $\tau(j)-\tau(k)\notin I$ and analyze the two cases for not being in $I$:

$\boldsymbol{\tau(j)-\tau(k)<m}$: $\boldsymbol{m}^{\langle h,t\rangle}(t_k-t_j\leq -m)=0$ since $\tau(k)=h(t_k)$ and $\tau(j)=h(t_j)$. The three-valued semantics fo $\mathrm{HT}_c$ gives us that $\boldsymbol{m}^{\langle h,t\rangle}(\neg(t_k-t_j\leq -m))=2$. Since $\langle h,t\rangle\models \boldsymbol{\Psi}^c_\lambda(P)$ then $\boldsymbol{m}^{\langle h,t\rangle}(\delta^{k,j}_{[m..n)}\leftarrow\neg(t_k-t_j\leq -m))=2$, which leads to $\boldsymbol{m}^{\langle h,t\rangle}(\delta^{k,j}_I=\boldsymbol{t})=2$ using the three-valued semantics for $\mathrm{HT}_c$.

$\boldsymbol{\tau(j)-\tau(k)\geq n}$: Analogous

$\rightarrow$: We start with $\boldsymbol{m}^{\langle h,t\rangle}(\delta^{k,j}_I=\boldsymbol{t})=2$, and let's assume towards a contradiction that $\tau(j)-\tau(k)\in I$. Then, $\boldsymbol{m}^{\langle h,t\rangle}(t_k-t_j\leq -m)=2$ and $\boldsymbol{m}^{\langle h,t\rangle}(t_j-t_k\leq n-1)=2$. Using the three-valued semantics for $\mathrm{HT}_c$ we can put together the following conjunction: $\boldsymbol{m}^{\langle h,t\rangle}(\delta^{k,j}_I\wedge(t_k-t_j\leq -m)\wedge(t_j-t_k\leq n-1))=2$, and its corresponding constraint, $\boldsymbol{m}^{\langle h,t\rangle}(\bot\leftarrow\delta^{k,j}_I\wedge(t_k-t_j\leq -m)\wedge(t_j-t_k\leq n-1))=0$. This leads to a contradiction since $\langle h,t\rangle\models\boldsymbol{\Psi}^{c\bot}_\lambda(P)$.

We will now show that $\boldsymbol{m}^{\langle h,t\rangle}(\delta^{k,j}_I=\boldsymbol{t})=0$ iff $\tau(j)-\tau(k)\in I$ for all $k\in[0..\lambda)$, $I\in\mathbb{I}$. For this let $k\in[0..\lambda)$ and $I=[m..n)\in\mathbb{I}$. As before, we know that $h(t_k)=\tau(k)$ and $h(t_j)=\tau(j)$. Now we prove both directions of the equivalence:

$\leftarrow$:
We start with $\tau(j)-\tau(k)\in I$. Then, $\boldsymbol{m}^{\langle h,t\rangle}(t_k-t_j\leq -m)=2$ and $\boldsymbol{m}^{\langle h,t\rangle}(t_j-t_k\leq n-1)=2$. Since $\langle h,t\rangle\models\boldsymbol{\Psi}^{c\bot}_\lambda(P)$, then $\boldsymbol{m}^{\langle h,t\rangle}(\bot\leftarrow\delta^{k,j}_I\wedge(t_k-t_j\leq -m)\wedge(t_j-t_k\leq n-1))=2$, which means that $\boldsymbol{m}^{\langle h,t\rangle}(\delta^{k,j}_I\wedge(t_k-t_j\leq -m)\wedge(t_j-t_k\leq n-1))=0$ By substituting the know values the three-valued semantics for $\mathrm{HT}_c$ gives us that $\min\{\boldsymbol{m}^{\langle h,t\rangle}(\delta^{k,j}_{[m..n)}),2\}=0$. Therefore, $\boldsymbol{m}^{\langle h,t\rangle}(\delta^{k,j}_I=\boldsymbol{t})=0$.

$\rightarrow$:
We start with $\boldsymbol{m}^{\langle h,t\rangle}(\delta^{k,j}_I=\boldsymbol{t})=0$, and assume towards a contradiction that $\tau(j)-\tau(k)\notin I$. We analyze the two cases for not being in $I$:

$\boldsymbol{\tau(j)-\tau(k)<m}$: We have that $\boldsymbol{m}^{\langle h,t\rangle}(t_k-t_j\leq -m)=0$ so $\boldsymbol{m}^{\langle h,t\rangle}(\neg(t_k-t_j\leq -m))=2$. Since $\langle h,t\rangle\models\boldsymbol{\Psi}^c_\lambda(P)$, then $\boldsymbol{m}^{\langle h,t\rangle}(\delta^{k,j}_{[m..n)}\leftarrow\neg(t_k-t_j\leq -m))=2$, which leads to $\boldsymbol{m}^{\langle h,t\rangle}(\delta^{k,j}_I=\boldsymbol{t})=2$. This is a contradiction.

$\boldsymbol{\tau(j)-\tau(k)\geq n}$: Analogous

Then $\tau(j)-\tau(k)\in I$
We will now show that that $\boldsymbol{m}^{\langle h,t\rangle}(L^{k,j}_\mu)=\max\{\boldsymbol{m}^{\langle h,t\rangle}(L^j_b),\boldsymbol{m}^{\langle h,t\rangle}(\delta^{k,j}_I)\}$. Consider $\mu=\Box_I b$ and let $k\in[0..\lambda)$, $j\in[k..\lambda)$. Since $\langle h,t\rangle\models\boldsymbol{\Pi}_\lambda(P)$, we have that $\boldsymbol{m}^{\langle h,t\rangle}(\eta_k(\mu))=2$. We use the definition of Table 3 and get $\boldsymbol{m}^{\langle h,t\rangle}(L^{k,j}_\mu\leftrightarrow L^j_b\vee\delta^{k,j}_I)=2$. By applying Proposition 11, $\boldsymbol{m}^{\langle h,t\rangle}(L^{k,j}_\mu)=\boldsymbol{m}^{\langle h,t\rangle}(L^j_b\vee\delta^{k,j}_I)$, which by the semantics of three-valued logic gives us that $\boldsymbol{m}^{\langle h,t\rangle}(L^{k,j}_\mu)=\max\{\boldsymbol{m}^{\langle h,t\rangle}(L^j_b),\boldsymbol{m}^{\langle h,t\rangle}(\delta^{k,j}_I)\}$. The prove that $\boldsymbol{m}^{\langle h,t\rangle}(L^{k,j}_\mu)=\min\{\boldsymbol{m}^{\langle h,t\rangle}(L^j_b),\boldsymbol{m}^{\langle h,t\rangle}(\neg\delta^{k,j}_I)\}$, follows analogously. □

*Theorem 8 (Completeness).*
Let $P$ be a metric logic program over $\mathcal{A}$, and $\mathbf{M}=(\langle\mathbf{T},\mathbf{T}\rangle,\tau)$ a total timed HT-trace of length $\lambda$.

Downloaded from https://www.cambridge.org/core. UNIVERSIDAD AUTONOMA METROPOLITANA, on 08 Nov 2026 at 14:47:57, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://doi.org/10.1017/S1471068426100714

If $\mathbf{M}$ is a metric equilibrium model of $P$, then there exists a constraint equilibrium model $\langle t, t\rangle$ of $\mathbf{\Pi}_\lambda(P) \cup \Delta^c_\lambda \cup \mathbf{\Psi}^c_\lambda(P)$ such that $\langle t, t\rangle|_{\mathcal{A}^* \cap \mathcal{T}^c} = \theta^c(\mathbf{M})$.

*Proof.*
Let $\boldsymbol{m}^\tau$ be the three-valued metric interpretation $\boldsymbol{m}^\tau$ in correspondence with $\mathbf{M}$ (Definition 10). We construct $\langle t, t\rangle = \chi(\boldsymbol{m}^\tau, P)$. We notice the properties from Proposition 24 are satisfied by the construction of $\chi(\boldsymbol{m}^\tau, P)$, so we can apply Proposition 24 to assure that $\langle t, t\rangle|_{\mathcal{A}^* \cup \mathcal{T}^c} = \theta^c(\mathbf{M})$. We also know that $\chi(\boldsymbol{m}^\tau, P)$ is total by Proposition 26 since $\mathbf{M}$ is total. In what follows, we will prove the theorem by proving incrementally that $\langle t, t\rangle$ is a constraint equilibrium model of each part of the program using Lemma 15, limiting the alphabet to the corresponding atoms.

We begin proving that that $\langle t, t\rangle|_{\mathcal{T}^c}$ is an equilibrium model of $\Delta^c_\lambda$. Notice that by construction $\langle t, t\rangle$ is timed wrt. $\lambda$ inducing $\tau$. Therefore, $\langle t, t\rangle|_{\mathcal{T}^c}$ is a constraint equilibrium model of $\Delta^c_\lambda$ by applying Proposition 20.

We will now prove that $\langle t, t\rangle|_{\mathcal{T}^c \cup \mathfrak{I}}$ is an equilibrium model of $\Delta^c_\lambda \cup \mathbf{\Psi}^c_\lambda(P)$ We can prove that $\langle t, t\rangle|_{\mathcal{T}^c \cup \mathfrak{I}} \models \mathbf{\Psi}^c_\lambda(P)$ as in the prove of Lemma 17. For the equilibrium criteria, we assume towards a contradiction that there is $\langle h, t\rangle$ such that $h \subset t$ and $\langle h, t\rangle \models \mathbf{\Psi}^c_\lambda(P)$. This means that there is $x \in \mathfrak{I}$ such that $h(x) = \boldsymbol{u}$ and $t(x) = \boldsymbol{t}$ (Notice that $\langle h, t\rangle$ is still timed). Then, $x = \delta^{k,j}_I$ for some $k \in [0..\lambda)$, $j \in [k..\lambda)$ and $I \in \mathbb{I}$. Since $x \in t$, $\boldsymbol{m}^{\chi(\boldsymbol{m}^\tau, P)}(\delta^{k,j}_I = \boldsymbol{t}) = 2$. By construction of $\boldsymbol{m}^{\chi(\boldsymbol{m}^\tau, P)}$ $\tau(j) - \tau(k) \notin I$. Now we analyze each case for not being in $I$:

$\boldsymbol{\tau(j) - \tau(k) < m}$: Then, $\boldsymbol{m}^{\langle h,t\rangle}(t_k - t_j \leq -m) = 0$, since $\tau(k) = h(t_k)$ and $\tau(j) = h(t_j)$. By $\mathrm{HT}_c$ three-valued semantics we know that $\boldsymbol{m}^{\langle h,t\rangle}(\neg(t_k - t_j \leq -m)) = 2$, and since $\boldsymbol{m}^{\langle h,t\rangle}(\delta^{k,j}_{[m..n)} \leftarrow \neg(t_k - t_j \leq -m)) = 2$, we have that $\boldsymbol{m}^{\langle h,t\rangle}(\delta^{k,j}_I = \boldsymbol{t}) = 2$, which leads to a contradiction since $h(x) = \boldsymbol{u}$

$\boldsymbol{\tau(j) - \tau(k) \geq n}$: Analogous

Therefore, $\langle t, t\rangle|_{\mathcal{T}^c \cup \mathfrak{I}}$ is a constraint equilibrium model of $\Delta^c_\lambda \cup \mathbf{\Psi}^c_\lambda(P)$.

We will prove that $\langle t, t\rangle$ is a constraint equilibrium model of $\Delta^c_\lambda \cup \mathbf{\Psi}^c_\lambda(P) \cup \mathbf{\Pi}_\lambda(P)$. We can prove that $\langle t, t\rangle \models \mathbf{\Pi}_\lambda(P)$ as in the proof of Lemma 17. For the equilibrium criteria, we assume towards a contradiction that there is $\langle h, t\rangle$ such that $h \subset t$ and $\langle h, t\rangle \models \mathbf{\Pi}_\lambda(P) \cup \Delta^c_\lambda \cup \mathbf{\Psi}^c_\lambda(P)$. Then, there is $x \in \mathcal{A}^* \cup \mathcal{L}_P$ such that $(x, \boldsymbol{t}) \in t$ and $(x, \boldsymbol{t}) \notin h$ so $\boldsymbol{m}^{\langle h,t\rangle}(x = \boldsymbol{t}) = 1$. We know that it can't be $x \in \mathcal{T}^c \cup \mathfrak{I}$ since $\langle t, t\rangle|_{\mathcal{T}^c \cup \mathfrak{I}} = \langle h, t\rangle|_{\mathcal{T}^c \cup \mathfrak{I}}$.

$\boldsymbol{x \in \mathcal{A}^*}$: We have that $x = a_k$ for some $0 \leq k < \lambda$. We get $\langle t, t\rangle|_{\mathcal{T}^c \cup \mathfrak{I}} \models \mathbf{\Psi}^{\perp,c}_\lambda(P)$ by applying Lemma 16 since $\langle t, t\rangle|_{\mathcal{T}^c \cup \mathfrak{I}}$ is an equilibrium model of $\Delta^c_\lambda \cup \mathbf{\Psi}^c_\lambda(P)$. Notice that, since $\langle h, t\rangle|_{\mathcal{T}^c \cup \mathfrak{I}} = \langle t, t\rangle|_{\mathcal{T}^c \cup \mathfrak{I}}$, we have $\langle h, t\rangle \models \mathbf{\Psi}^{\perp,c}_\lambda(P)$. From $\langle h, t\rangle \models \mathbf{\Pi}_\lambda(P) \cup \mathbf{\Psi}^c_\lambda(P)$ and Lemma 18, we know that $\sigma^c(\langle h, t\rangle|_{\mathcal{A}^* \cup \mathcal{T}^c}) \models P$. Which leads to a contradiction since $\mathbf{M}$ is an equilibrium model of $P$ but $\sigma^c(\langle h, t\rangle|_{\mathcal{A}^* \cup \mathcal{T}^c}) < \mathbf{M}$. This means that $\langle t, t\rangle|_{\mathcal{A}^*} = \langle h, t\rangle|_{\mathcal{A}^*}$.

$\boldsymbol{x \in \mathcal{L}_P}$: Notice that if $\langle h, t\rangle \models \mathbf{\Pi}_\lambda(P)$ then the value of $x \in \mathcal{L}_P$ is fully defined by the atoms in $\mathcal{A}^* \cup \mathfrak{I}$. Since $\langle t, t\rangle|_{\mathcal{A}^* \cup \mathfrak{I}} = \langle h, t\rangle|_{\mathcal{A}^* \cup \mathfrak{I}}$, then, $\boldsymbol{m}^{\langle h,t\rangle}(x) = \boldsymbol{m}^{\langle t,t\rangle}(x)$. But since $\langle t, t\rangle$ is total then $\boldsymbol{m}^{\langle t,t\rangle}(x)$ is either 0 or 1, but $\boldsymbol{m}^{\langle h,t\rangle}(x) = 1$. Which leads to a contradiction.

Therefore, $\langle t, t\rangle$ is a constraint equilibrium model of $\Delta^c_\lambda \cup \mathbf{\Psi}^c_\lambda(P) \cup \mathbf{\Pi}_\lambda(P)$. □

Downloaded from https://www.cambridge.org/core. UNIVERSIDAD AUTONOMA METROPOLITANA, on 08 Nov 2026 at 14:47:57, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://doi.org/10.1017/S1471068426100714

*Theorem 9 (Correctness).*
Let $P$ be a metric logic program, and $\lambda \in \mathbb{N}$.

If $\langle t, t\rangle$ is an constraint equilibrium model of $\mathbf{\Pi}_\lambda(P) \cup \Delta_\lambda^c \cup \mathbf{\Psi}_\lambda^c(P)$, then $\sigma^c(\langle t, t\rangle|_{\mathcal{A}^* \cap \mathcal{T}^c})$ is a metric equilibrium model of $P$.

*Proof.*
Let $\mathbf{M} = \sigma^c(\langle t, t\rangle|_{\mathcal{A}^* \cap \mathcal{T}^c})$.

We begin proving that $\mathbf{M} \models P$. We do so by using Lemma 18, so it remains proving that the conditions for this Lemma hold. Firstly, $\langle t, t\rangle|_{\mathcal{T}^c}$ is a constraint equilibrium model of $\Delta_\lambda^c$ due to Lemma 15. Due to Proposition 4, $\langle t, t\rangle|_{\mathcal{T}^c}$ is timed wrt. $\lambda$ inducing $\tau$. Since adding other atoms not in $\mathcal{T}^c$ keeps timed property, $\langle t, t\rangle$ is also timed wrt. $\lambda$ inducing $\tau$. For the second condition. $\langle t, t\rangle|_{\mathcal{T}^c \cup \mathfrak{I}}$ is a constraint equilibrium model of $\Delta_\lambda^c \cup \mathbf{\Psi}_\lambda^c(P)$ thanks to the Splitting property. Applying Lemma 16 we get that $\langle t, t\rangle|_{\mathcal{T}^c \cup \mathfrak{I}} \models \mathbf{\Psi}_\lambda^\perp(P)$. As before, we can add the rest of the atoms since there is no interference so $\langle t, t\rangle \models \mathbf{\Psi}_\lambda^\perp(P)$. With this we can apply Lemma 18 to get that $\mathbf{M} \models P$.

We then prove that it is in equilibrium. By construction $\mathbf{M}$ is total. We assume towards a contradiction that there is $\mathbf{M}' = (\langle \mathbf{H}, \mathbf{T}\rangle, \tau)$ such that $\mathbf{H} < \mathbf{T}$ and $\mathbf{M}' \models P$ Then, there is $a \in \mathcal{A}$ such that $a \in T_k$ and $a \notin H_k$, for some $0 \leq k < \lambda$. We apply Lemma 17, and let $\langle h, t\rangle$ be the $\mathrm{HT}_c$ interpretation such that $\langle h, t\rangle \models \mathbf{\Pi}_\lambda(P) \cup \Delta_\lambda^c \cup \mathbf{\Psi}_\lambda^c(P)$ with $\nu = \tau(\lambda - 1)$, and $\langle H, T\rangle|_{\mathcal{A}^* \cup \mathcal{T}^c} = \theta(\mathbf{M}')$. By construction $(a_k, \boldsymbol{t}) \in t$ and $(a_k, \boldsymbol{t}) \notin h$, which means that $\langle h, t\rangle < \langle t, t\rangle$ leading to a contradiction since $\langle t, t\rangle$ is a constraint equilibrium model. Since there is no smaller model, $\mathbf{M}$ is in equilibrium. □

Downloaded from https://www.cambridge.org/core. UNIVERSIDAD AUTONOMA METROPOLITANA, on 08 Nov 2026 at 14:47:57, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://doi.org/10.1017/S1471068426100714