Explicit Negation in Linear-Dynamic Equilibrium Logic

Felicidad Aguado[1] **Pedro Cabalar**,[1] **Jorge Fandinno**[2] **Gilberto Pérez**[1] and **Concepción Vidal**[1]

Abstract. In this paper, we revisit a temporal extension of Equilibrium Logic (the logical characterisation of Answer Set Programming) that introduces Linear Dynamic Logic modalities. In particular, we further incorporate to this extension (we call Linear Dynamic Equilibrium Logic) an explicit negation operator, treated as a regular logical connective. We explain several formal properties of this new extension. For instance, we prove that some temporal operators that were not inter-definable, become so if we allow the use of explicit negation. Finally, we also introduce and study a new temporal operator called "while," that is an implicational dual of "until" and may be useful as a basic connective for temporal logic programming.

1 Introduction

Based on the answer set (or stable model) semantics [12, 13] for logic programs, *Answer Set Programming* [17, 18] (ASP) has become one of the most successful paradigms for practical Knowledge Representation and problem solving. Although ASP is naturally equipped for solving static combinatorial problems up to NP complexity (or Σ_2^P in the disjunctive case) its application to temporal scenarios has been frequent since its very beginning, partly due to its early use for reasoning about actions and change [14]. Temporal problems normally suppose an extra challenge for ASP for several reasons. On the one hand, they normally raise the complexity (in the case of classical planning, for instance, it becomes PSPACE-complete [6]), although this is usually accounted for by making repeated calls to an ASP solver. On the other hand, temporal scenarios also pose a representational challenge, since the basic ASP language does not support temporal expressions.

To fill this representational gap, a temporal extension of ASP called *Temporal Equilibrium Logic* (TEL) was proposed in [10] and extensively studied later on [1]. This formalism constitutes a modal linear-time extension of *Equilibrium Logic* [19] which, in its turn, is a complete logical characterisation of (standard) ASP based on the intermediate logic of *Here-and-There* (HT) [16]. In a recent line of research [9], TEL was extended to cope with finite traces (which are closer to ASP computation), leading to an implementation of a first temporal ASP solver, `telingo` [8]. Finally, following similar steps to [11], where the relation between Linear-Time Temporal Logic (LTL) [20] and Linear Dynamic Logic (LDL) for finite traces was studied, an LDL extension of ASP was analogously introduced in [5, 7]. This latest extension, called *Linear Dynamic Equilibrium Logic* (DEL), essentially introduces dynamic logic modalities that allow for describing temporal paths in terms of regular expressions. To put an example, the formula $[\neg help^*](\neg help \rightarrow sos)$ behaves as a logic program rule that repeats sending an sos while no evidence of $help$ has been received along a sequence of states. DEL is general enough to cover LDL, as it shares the same syntax but introduces non-monotonicity with the definition of temporal stable models. It also covers LTL and TEL as particular cases, since LTL temporal operators can be defined as particular cases of DEL expressions: for instance $\Box\alpha$ (i.e. α always holds) can be represented in DEL as $[\top^*]\alpha$. Despite of this generality and as a consequence of its novelty, many features of DEL are still unexplored. In the implementation side, an extension of `telingo` to incorporate dynamic logic expressions is being developed. On the theoretical ground, however, there are still many open representational issues such as expressiveness in comparison to TEL or in combination with other ASP extensions.

In this paper, we study one of such combinations that has not been tackled so far: the incorporation of the *explicit negation* operator (as a regular logical connective) into temporal ASP. Both in ASP and in its logical counterpart, Equilibrium Logic, the standard negation operator stands for *default negation*, that is, $\neg p$ represents that "there is no evidence about p." However, in many ASP scenarios, and in most of those related to reasoning about actions, we frequently find a second negation operator, we call *explicit negation* and denote as '$\sim$'. This second negation represents explicit falsity so that $\sim p$ means that "there is evidence about the falsity of p." Explicit negation was first introduced[3] in [13] and is extensively used nowadays in ASP, although only applied to atoms. Its treatment as a logical connective was first proposed with the definition of Equilibrium Logic [19] and was recently revised in [2] for a better behaviour with respect to program reduct transformations. For instance, using this operator, we may not only write rules like $\neg guilty \rightarrow \sim guilty$, meaning that $guilty$ is explicitly false by default, but also nest $\sim$in an arbitrary way, as in $\sim(money \wedge time) \rightarrow \sim travel$ meaning that I do not travel if I do not have time and money. This last expression is actually equivalent to the pair of rules $\sim money \rightarrow \sim travel$ and $\sim time \rightarrow \sim travel$. Although [2] provides a full interpretation based on stable models for arbitrary propositional theories with explicit negation, the use of this operator in temporal ASP has been completely unexplored up to date. In this paper, we provide an extension of DEL for incorporating explicit negation and study its properties. Since TEL can be defined as a particular case of DEL, we also provide a semantics for explicitly negated linear-time temporal operators. For instance, we prove De Morgan-style properties and show that $\sim\Diamond danger$ is actually equivalent to $\Box \sim danger$, that is, $danger$ is always explicitly false. Moreover, we prove that explicit negation allows some inter-definability of operators like $\Box p \equiv \sim\Diamond \sim p$ something well-known to fail in TEL when using default negation, that is, $\Box p \not\equiv \neg\Diamond\neg p$. In the last part of the paper, as a result emerged from the study of DEL, we introduce a new temporal operator for TEL called *while* and somehow dual to the standard *until* from LTL.

This paper is organised as follows. The next section introduces the syntax and semantics of DEL with explicit negation, defining tempo-

[1] IRLab / CITIC Research Center, University of A Coruña, Spain. emails: {aguado,cabalar,gperez,eicovima}@udc.es

[2] University of Potsdam, Germany. email: fandinno@uni-potsdam.de

[3] Called there *classical* negation.

ral operators from LTL as abbreviations. This section also explains the different types of equivalences that arise and provides several fundamental properties. Section 3 is focused on the *while* operator and provides properties that characterise its relation to *until* and *release*. Finally, Section 4 concludes the paper.

2 Syntax and semantics

Given a set $\mathcal{A}$ of propositional variables (called *alphabet*), the grammar rules for *dynamic formulas* φ and *path expressions* ρ are mutually defined as in [11], but adding here the explicit negation operator $\sim$in the following way:

$$\begin{array}{rcl}\varphi & ::= & a \mid \bot \mid \top \mid \; [\rho]\,\varphi \;\mid\; \langle\rho\rangle\,\varphi \;\mid {\sim}\varphi \\ \rho & ::= & \tau \mid \varphi? \mid \rho+\rho \mid \rho\,;\rho \mid \rho^* \mid \rho^-\end{array}$$

Each ρ is a regular expression formed with the path constant τ (read as "step") plus the usual test construct $\varphi?$ from Dynamic Logic (DL [15]) and the converse operator ρ^- for switching the temporal orientation from future to past and vice versa. A path expression not containing any test construct $\varphi?$ is said to be *test-free*. Note that, as in [11], we depart from DL, where atomic path expressions are *actions* from a different sort from propositional atoms. Here, we only define one atomic expression, τ, but will allow using Boolean formulas for that role too, introducing them through an abbreviation. Still, the reader may have noticed that formulas do not include Boolean operators. This is because they can be actually defined in terms of the necessity and possibility modalities in the following way:

$$\begin{array}{rclcrcl}\varphi\wedge\psi & \stackrel{def}{=} & \langle\varphi?\rangle\,\psi & \qquad & \varphi\vee\psi & \stackrel{def}{=} & \langle\varphi?+\psi?\rangle\,\top \\ \varphi\rightarrow\psi & \stackrel{def}{=} & [\varphi?]\,\psi & & \neg\varphi & \stackrel{def}{=} & \varphi\rightarrow\bot\end{array}$$

Double implication $\varphi \leftrightarrow \psi$ is defined in the usual way as $(\varphi \rightarrow \psi) \wedge (\psi \rightarrow \varphi)$. Note that, while conjunction is expressed in terms of possibility $\langle\varphi?\rangle\,\psi$, its dual for the necessity operator $[\varphi?]\,\psi$ is not a disjunction, but an *implication* instead. This is an important feature, since implication in intuitionistic and intermediate logics (like HT) is an elementary Boolean connective that cannot be defined in terms of the others. Note also that default negation $\neg\varphi$ follows the standard definition from intuitionistic negation in terms of implication $\varphi \rightarrow \bot$ which, in our case, amounts to $[\varphi?]\,\bot$. We say that a formula is *propositional* if it only contains combinations of $\wedge, \vee, \rightarrow, \neg, \bot, \top, \sim$ and atoms. We will allow using any propositional formula ϕ as a path expression standing for $(\phi?;\tau)$, something also done in LDL [11]. In particular, this allows us using $\top$ as the path expression $(\top?;\tau)$ which, as we see below, amounts to τ. Another abbreviation we use is the sequence of n repetitions of some expression ρ defined as $\rho^0 \stackrel{def}{=} \tau?$ and $\rho^{n+1} \stackrel{def}{=} \rho;\rho^n$. For instance, $\rho^3 = \rho;\rho;\rho;\tau?$ which amounts to $\rho;\rho;\rho$, as we will see when describing the semantics. We sometimes use $\rho^+ \stackrel{def}{=} \rho;\rho^*$, that is, repeating ρ at least once.

As with Boolean connectives, LTL modal connectives can be defined as derived operators as follows:

$$\begin{array}{rclcrcl}\circ\varphi & \stackrel{def}{=} & \langle\tau\rangle\,\varphi & \qquad & \widehat{\circ}\varphi & \stackrel{def}{=} & [\tau]\,\varphi \\ \Diamond\varphi & \stackrel{def}{=} & \langle\tau^*\rangle\,\varphi & & \Box\varphi & \stackrel{def}{=} & [\tau^*]\,\varphi \\ \varphi\,\mathbf{U}\,\psi & \stackrel{def}{=} & \langle(\varphi?;\tau)^*\rangle\,\psi & & \varphi\,\mathbf{W}\,\psi & \stackrel{def}{=} & [(\psi?;\tau)^*]\,\varphi \\ \varphi\,\mathbf{R}\,\psi & \stackrel{def}{=} & (\psi\,\mathbf{U}\,(\varphi\wedge\psi))\vee\Box\psi & & \mathbf{F} & \stackrel{def}{=} & \widehat{\circ}\bot\end{array}$$

Operators $\circ$, $\Diamond$, $\Box$, $\mathbf{U}$, $\mathbf{R}$ are the standard *next, eventually, always, until* and *release* from LTL. Operator $\widehat{\circ}$ is the *weak* dual of *next* and is relevant for finite traces: $\widehat{\circ}\varphi$ means that φ holds in the next state, if there is a next state. Using the weak next we can define, for instance, formula $\mathbf{F}$ as $\widehat{\circ}\bot = [\tau]\,\bot$ that holds when we are at the *final* situation of a (finite) trace. The expression $\varphi\,\mathbf{W}\,\psi$ is new: it corresponds to the genuine necessity-dual[4] of *until* and it is read as "repeat φ *while* ψ holds." Note how this formula in DEL is different from *release*, which is the standard dual operator of *until* in LTL. We discuss this new operator in detail in Section 3. Analogous past-oriented operators can be defined by replacing above any path expression ρ by its converse ρ^-. For instance $\bullet\varphi \stackrel{def}{=} \langle\tau^-\rangle\,\varphi$ would correspond to the *previous* operator. For simplicity, we omit past-oriented temporal formulas in this paper since all the properties studied here are trivially extrapolated to that case.

A formula is *temporal*, if it includes only Boolean and temporal operators. A dynamic formula is said to be *conditional* if it contains some occurrence of an atom $p \in \mathcal{A}$ inside a $[\cdot]$ operator; it is called *unconditional* otherwise. Note that formulas with atoms in implication antecedents or negated formulas are also conditional, since they are derived from $[\cdot]$. For instance, $[p?]\,\bot$ is conditional, and is actually the same as $p \rightarrow \bot$ and $\neg p$. As usual, a *(dynamic) theory* is a set of (dynamic) formulas.

For the semantics, we rely on the idea of linear sequences of states, called *traces*. An *explicit literal* is a formula just formed by an atom $a \in \mathcal{A}$ or its explicit negation $\sim a$. A *state* H over alphabet $\mathcal{A}$ is a consistent set of explicit literals for atoms in $\mathcal{A}$, that is, there is no atom $p \in \mathcal{A}$ for which $\{p, \sim p\} \subseteq H$. Now, for denoting intervals of time points we use the following notation. Given $x \in \mathbb{N}$ and $y \in \mathbb{N} \cup \{\omega\}$, we let $[x..y]$ stand for the set $\{i \in \mathbb{N} \mid x \leq i \leq y\}$ and $[x..y)$ for $\{i \in \mathbb{N} \mid x \leq i < y\}$. A *trace* $\mathbf{H}$ of length λ over alphabet $\mathcal{A}$ is a sequence of states $\mathbf{H} = \langle H_i\rangle_{i\in[0..\lambda)}$. Trace $\mathbf{H}$ is *infinite* if $\lambda = \omega$ and *finite* otherwise, that is, $\lambda \in \mathbb{N}$. Given traces $\mathbf{H} = \langle H_i\rangle_{i\in[0..\lambda)}$ and $\mathbf{H}' = \langle H'_i\rangle_{i\in[0..\lambda)}$ both of length λ, we write $\mathbf{H} \leq \mathbf{H}'$ if $H_i \subseteq H'_i$ for each $i \in [0..\lambda)$; accordingly, $\mathbf{H} < \mathbf{H}'$ iff both $\mathbf{H} \leq \mathbf{H}'$ and $\mathbf{H} \neq \mathbf{H}'$.

A *Here-and-There trace* (for short HT-*trace*) of length λ over alphabet $\mathcal{A}$ is a sequence of pairs $\langle H_i, T_i\rangle_{i\in[0..\lambda)}$ such that $H_i \subseteq T_i$ are states for any $i \in [0..\lambda)$. As before, an HT-trace is infinite if $\lambda = \omega$ and finite otherwise. We can simply represent an HT-trace as a pair of traces $\langle\mathbf{H}, \mathbf{T}\rangle$ of length λ where $\mathbf{H} = \langle H_i\rangle_{i\in[0..\lambda)}$ and $\mathbf{T} = \langle T_i\rangle_{i\in[0..\lambda)}$ and $\mathbf{H} \leq \mathbf{T}$. A particular type of HT-traces satisfy $\mathbf{H} = \mathbf{T}$ and are called *total*. The intuition of using these two traces is the same from HT and Equilibrium Logic: explicit literals in H_i are those that can be proved at time point i; explicit literals not in T_i are those at i for which there is no proof; and, finally, explicit literals in $T_i \setminus H_i$ are assumed to be true, but have not been proved.

We proceed to extend the linear dynamic logic of HT (DHT) presented in [7] to cope with explicit negation. This will be achieved by additionally providing a falsification relation $=\!\mid$ dual to DHT regular satisfaction $\models$. Both relations are defined on a double induction. Given any HT-trace $\mathbf{M} = \langle\mathbf{H}, \mathbf{T}\rangle$, we define DHT satisfaction (falsification) of formulas, $\mathbf{M}, k \models \varphi$ ($\mathbf{M}, k =\!\mid \varphi$), in terms of two accessibility relations for path expressions $\|\rho\|^{\mathbf{M}}_{\models} \subseteq \mathbb{N}^2$ and $\|\rho\|^{\mathbf{M}}_{=\!\mid} \subseteq \mathbb{N}^2$ whose extents depend on $\models$ and $=\!\mid$.

Definition 1 (DHT **satisfaction / falsification)** *An* HT-*trace* $\mathbf{M} = \langle\mathbf{H}, \mathbf{T}\rangle$ *of length* λ *over alphabet* $\mathcal{A}$ satisfies *a dynamic formula* φ *at time point* $k \in [0..\lambda)$ *written* $\mathbf{M}, k \models \varphi$*, if the following conditions hold:*

1. $\mathbf{M}, k \models \top$ *and* $\mathbf{M}, k \not\models \bot$

[4] We actually changed the order of φ and ψ with respect to *until* to facilitate a more natural reading of *while*.

2. $\mathbf{M}, k \models a$ *if* $a \in H_k$ *for any atom* $a \in \mathcal{A}$
3. $\mathbf{M}, k \models \langle\rho\rangle\,\varphi$ *if* $\mathbf{M}, i \models \varphi$ *for some* i *with* $(k,i) \in \|\rho\|^{\mathbf{M}}_{\models}$
4. $\mathbf{M}, k \models [\rho]\,\varphi$ *if* $\mathbf{M}', i \models \varphi$ *for all* i *with* $(k,i) \in \|\rho\|^{\mathbf{M}'}_{\models}$ *for both* $\mathbf{M}' = \mathbf{M}$ *and* $\mathbf{M}' = \langle \mathbf{T}, \mathbf{T}\rangle$
5. $\mathbf{M}, k \models {\sim}\varphi$ *if* $\mathbf{M}, k =\!\!| \varphi$

An HT*-trace* $\mathbf{M} = \langle \mathbf{H}, \mathbf{T}\rangle$ *of length* λ *over alphabet* $\mathcal{A}$ falsifies *a dynamic formula* φ *at time point* $k \in [0..\lambda)$ *written* $\mathbf{M}, k =\!\!| \varphi$*, if the following conditions hold:*

6. $\mathbf{M}, k \not=\!\!| \top$ *and* $\mathbf{M}, k =\!\!| \bot$
7. $\mathbf{M}, k =\!\!| {\sim} a$ *if* ${\sim} a \in H_k$ *for any atom* $a \in \mathcal{A}$
8. $\mathbf{M}, k =\!\!| \langle\rho\rangle\,\varphi$ *if* $\mathbf{M}, i =\!\!| \varphi$ *for all* i, $(k,i) \in \|\rho\|^{\mathbf{M}}_{\not=\!|}$
9. $\mathbf{M}, k =\!\!| [\rho]\,\varphi$ *if* $\mathbf{M}, i =\!\!| \varphi$ *for some* $(k,i) \in \|\rho\|^{\mathbf{M}'}_{\models}$ *where* $\mathbf{M}' = \langle \mathbf{T}, \mathbf{T}\rangle$
10. $\mathbf{M}, k =\!\!| {\sim}\varphi$ *if* $\mathbf{M}, k \models \varphi$

where, for any HT*-trace* $\mathbf{M}$, $\|\rho\|^{\mathbf{M}}_{\models} \subseteq \mathbb{N}^2$ *and* $\|\rho\|^{\mathbf{M}}_{\not=\!|} \subseteq \mathbb{N}^2$ *are two relations on pairs of time points inductively defined as follows. For each relation* $R \in \{\models, \not=\!\!|\}$*:*

11. $\|\tau\|^{\mathbf{M}}_R \stackrel{def}{=} \{(i, i+1) \mid i, i+1 \in [0..\lambda)\}$
12. $\|\varphi?\|^{\mathbf{M}}_{\models} \stackrel{def}{=} \{(i,i) \mid \mathbf{M}, i \models \varphi\}$ *and* $\|\varphi?\|^{\mathbf{M}}_{\not=\!|} \stackrel{def}{=} \{(i,i) \mid \mathbf{M}, i \not=\!\!| \varphi\}$
13. $\|\rho_1 + \rho_2\|^{\mathbf{M}}_R \stackrel{def}{=} \|\rho_2\|^{\mathbf{M}}_R \cup \|\rho_2\|^{\mathbf{M}}_R$
14. $\|\rho_1\,;\rho_2\|^{\mathbf{M}}_R \stackrel{def}{=} \{(i,j) \mid (i,k) \in \|\rho_1\|^{\mathbf{M}}_R$ *and* $(k,j) \in \|\rho_2\|^{\mathbf{M}}_R$ *for some* $k\}$
15. $\|\rho^*\|^{\mathbf{M}}_R \stackrel{def}{=} \bigcup_{n\geq 0} \|\rho^n\|^{\mathbf{M}}_R$
16. $\|\rho^-\|^{\mathbf{M}}_R \stackrel{def}{=} \{(i,j) \mid (j,i) \in \|\rho\|^{\mathbf{M}}_R\}$ □

Conditions 1-4 are the standard ones for DHT without explicit negation, as defined in [7]. The main difference here with respect to LDL is that the necessity operator must be checked both in the $\mathbf{H}$ and the $\mathbf{T}$ trace. This is related to the strong relation between $[\cdot]$ and intuitionistic implication since, as we saw before, $\varphi \to \psi$ is actually defined as $[\varphi?]\,\psi$. Condition 5 and its dual 10 assert that the $\sim$ operator just switches from satisfaction to falsification and vice versa. Then, conditions from 6-9 are duals of 1-4. In the case of 6 and 7, this duality is straightforward. Falsification of $\langle\rho\rangle\,\varphi$ is also quite natural: we replace $\models$ by $=\!\!|$ and change the quantification over i from existential to universal. The only special requirement here is that the pairs (k,i) for interpreting ρ will use the $\not=\!\!|$ relation in $\|\rho\|^{\mathbf{M}}_{\not=\!|}$. This condition is required in order to obtain the expected interpretation for $\langle\varphi?\rangle\,\psi$ with respect to its reading as conjunction $\varphi \wedge \psi$, where by "expected" we mean the semantics defined in [2] for explicit negation in propositional formulas. This means that $\mathbf{M}, k =\!\!| \varphi \wedge \psi$ should amount to $\mathbf{M}, k =\!\!| \varphi$ or $\mathbf{M}, k =\!\!| \psi$. From 7 and its duality to 2, the condition we get for $\mathbf{M}, k =\!\!| \langle\varphi?\rangle\,\psi$ must have the form $\mathbf{M}, i =\!\!| \psi$ for all i, $(k,i) \in \|\varphi?\|^{\mathbf{M}}_R$ for some relation R. Since the test just forces $i = k$ if relation R holds, we would get: $\mathbf{M}, k =\!\!| \psi$ or $\mathbf{M}, k\ R\ \varphi$ does not hold. Thus, to get the expected result, the only possibility is taking R to be $\not=\!\!|$. Condition 9 is also limited by the expected behaviour of explicit negation of implication which, in its turn, was the main reason to introduce the explicit negation variant in [2]. Following that paper, $\mathbf{M}, k =\!\!| \varphi \to \psi$ should be equivalent to $\mathbf{M}, k =\!\!| \psi$ and $\mathbf{M}', k \models \varphi$ with $\mathbf{M}' = \langle \mathbf{T}, \mathbf{T}\rangle$. This was proved there to preserve the expected behaviour with respect to reduct-based syntactic transformations similar to the original Gelfond-Lifschitz reduct [12]. In our context, falsifying implication corresponds to $\mathbf{M}, k =\!\!| [\varphi?]\,\psi$ and we can see that, under Condition 9, this amounts to make $i = k$ and assert $\mathbf{M}, k =\!\!| \psi$ and $(k,k) \in \|\rho\|^{\mathbf{M}'}_{\models}$. But this is equivalent to $\mathbf{M}, k =\!\!| \psi$ and $\mathbf{M}', k \models \varphi$ as we intended. The following theorem proves that the interpretation we obtain for satisfaction/falsification of derived Boolean connectives is exactly the one proposed in [2].

Theorem 1 *Let* $\mathbf{M} = \langle \mathbf{H}, \mathbf{T}\rangle$ *be an* HT*-trace of length* λ *over alphabet* $\mathcal{A}$ *and* $k \in [0..\lambda)$*. Given the respective definitions of derived operators, we get the following satisfaction and falsification conditions:*

1. $\mathbf{M}, k \models \alpha \wedge \beta$ *iff* $\mathbf{M}, k \models \alpha$ *and* $\mathbf{M}, k \models \beta$
2. $\mathbf{M}, k =\!\!| \alpha \wedge \beta$ *iff* $\mathbf{M}, k =\!\!| \alpha$ *or* $\mathbf{M}, k =\!\!| \beta$
3. $\mathbf{M}, k \models \alpha \vee \beta$ *iff* $\mathbf{M}, k \models \alpha$ *or* $\mathbf{M}, k \models \beta$
4. $\mathbf{M}, k =\!\!| \alpha \vee \beta$ *iff* $\mathbf{M}, k =\!\!| \alpha$ *and* $\mathbf{M}, k =\!\!| \beta$
5. $\mathbf{M}, k \models \alpha \to \beta$ *iff* $\mathbf{M}', k \not\models \alpha$ *or* $\mathbf{M}', k \models \beta$ *for both* $\mathbf{M}' = \mathbf{M}$ *and* $\mathbf{M}' = \langle \mathbf{T}, \mathbf{T}\rangle$
6. $\mathbf{M}, k =\!\!| \alpha \to \beta$ *iff* $\langle \mathbf{T}, \mathbf{T}\rangle, k \models \alpha$ *and* $\mathbf{M}, k =\!\!| \beta$

The next property guarantees that the interpretations of path expressions are always inside the time point limits (this is especially important for finite traces).

Proposition 1 *Relations* $\|\rho\|^{\mathbf{M}}_{\models}$ *and* $\|\rho\|^{\mathbf{M}}_{\not=\!|}$ *defined above satisfy* $\|\rho\|^{\mathbf{M}}_{\models} \subseteq [0..\lambda) \times [0..\lambda)$ *and* $\|\rho\|^{\mathbf{M}}_{\not=\!|} \subseteq [0..\lambda) \times [0..\lambda)$ *.* □

The following proposition relates $\|\rho\|^{\mathbf{M}}_{\models}$ and $\|\rho\|^{\mathbf{M}}_{\not=\!|}$ and shows an interesting property of test-free path expressions.

Proposition 2 *For any regular expression* ρ *and any HT-trace* $\mathbf{M}$*,*

$$\|\rho\|^{\mathbf{M}}_{\models} \subseteq \|\rho\|^{\mathbf{M}}_{\not=\!|}\,.$$

In addition, if ρ *is test-free, then*

$$\|\rho\|^{\mathbf{M}}_{\models} = \|\rho\|^{\mathbf{M}}_{\not=\!|} = \|\rho\|^{\mathbf{M}'}_{\models} = \|\rho\|^{\mathbf{M}'}_{\not=\!|}\,,$$

for any HT-traces $\mathbf{M} = \langle \mathbf{H}, \mathbf{T}\rangle$ *and* $\mathbf{M}' = \langle \mathbf{T}, \mathbf{T}\rangle$*.*

Proof . Notice that, when ρ is test-free, then $\|\rho\|^{\mathbf{M}}_{\models} = \|\rho\|^{\mathbf{M}}_{\not=\!|}$. On the other hand, if φ is any formula: $\|\varphi?\|^{\mathbf{M}}_{\models} \subseteq \|\varphi?\|^{\mathbf{M}}_{\not=\!|}$ because $\mathbf{M}, k \models \varphi$ implies $\mathbf{M}, k \not=\!\!| \varphi$ for any k. □

An HT-trace $\mathbf{M}$ is a *model* of a dynamic theory Γ if $\mathbf{M}, 0 \models \varphi$ for all $\varphi \in \Gamma$. We write $\mathrm{DHT}(\Gamma, \lambda)$ to stand for the set of DHT models of length λ of a theory Γ, and define $\mathrm{DHT}(\Gamma) \stackrel{def}{=} \bigcup_{\lambda=0}^{\omega} \mathrm{DHT}(\Gamma, \lambda)$, that is, the whole set of models of Γ of any length. When $\Gamma = \{\varphi\}$ we just write $\mathrm{DHT}(\varphi, \lambda)$ and $\mathrm{DHT}(\varphi)$.

A formula φ is a *tautology* (or is *valid*), written $\models \varphi$, iff $\mathbf{M}, k \models \varphi$ for any HT-trace $\mathbf{M}$ and any $k \in [0..\lambda)$. We call the logic induced by the set of all tautologies *(Linear) Dynamic logic of Here-and-There* (DHT for short).

We introduce non-monotonicity by defining the temporal version of stable models.

Definition 2 (Temporal equilibrium/stable model) *A* total THT*-trace* $\langle \mathbf{T}, \mathbf{T}\rangle$ *of length* λ *is a* temporal equilibrium model *of a theory* Γ *if* $\langle \mathbf{T}, \mathbf{T}\rangle \in \mathrm{DHT}(\Gamma, \lambda)$ *and there is no* $\langle \mathbf{H}, \mathbf{T}\rangle \in \mathrm{DHT}(\Gamma, \lambda)$ *with* $\mathbf{H} < \mathbf{T}$*. When this happens, we say that* $\mathbf{T}$ *is a* temporal stable model *of* Γ. □

The logic induced by temporal equilibrium models is called *Linear Dynamic Equilibrium Logic* (DEL). To illustrate the effect of DEL, suppose we wish to represent a scenario for breaking the glass of a fire extinguisher, but hitting the glass may arbitrarily fail. One possible representation could be the following theory:

$$[\tau^*; hit?]\,(broken \vee \neg broken) \quad (1)$$

$$[\tau^*; broken]\, broken \quad (2)$$

$$[\tau^*; \sim broken; (\neg broken)?]\ \sim broken \quad (3)$$

$$\sim broken \wedge [(\sim broken)^+]\, hit \quad (4)$$

The first formula, (1), means that hitting the glass may cause it to be broken. The expression $broken \vee \neg broken$ is not a tautology in DEL: it acts as a choice rule deriving the fact $broken$ or not. Formula (2) is an abbreviation[5] of $[\tau^*; broken?; \tau]\, broken$ and tells us that once the glass is broken, it remains so from then on. The implicative formula (3) is the inertia rule for being unbroken $\sim broken$: it is similar to (2) but adds the extra condition $(\neg broken)?$ meaning that we check that there is no evidence of $broken$ (using default negation) in the resulting state. Finally, (4) tells us that the glass is initially unbroken and that we will keep hitting it while it remains unbroken. This theory has temporal stable models satisfying either $\langle\ (hit \wedge \sim broken)^*; broken^*\ \rangle \top$ (that is, we succeeded to break the glass at some arbitrary time point) or $\langle\ (hit \wedge \sim broken)^*\ \rangle \top$ (that is, we hitted the glass forever without success). In the case of infinite traces, we could reject this last possibility including a fairness constraint of the form $[(\tau^*; hit \wedge \sim broken)^*]\perp$ so hitting without breaking the glass cannot occur infinitely often.

The introduction of explicit negation causes that a valid formula $\varphi \leftrightarrow \psi$ does not guarantee equivalence of the arbitrary substitution of φ by ψ in any context. We define, in fact, different types of equivalence. Two formulas φ, ψ are said to be *weakly equivalent*, written $\varphi \equiv_w \psi$, whenever $\mathbf{M}, k \models \varphi$ iff $\mathbf{M}, k \models \psi$ for any HT-trace $\mathbf{M}$ and any $k \in [0..\lambda)$. This is the same as requiring that $\varphi \leftrightarrow \psi$ is a tautology but does not mean that we can replace φ by ψ and vice versa in any context. For obtaining a congruence relation, we can use $\Leftrightarrow$ defined by $\varphi \Leftrightarrow \psi \stackrel{def}{=} (\varphi \leftrightarrow \psi) \wedge (\sim\varphi \leftrightarrow \sim\psi)$. Moreover, we say that two formulas φ, ψ are *equivalent*, written $\varphi \equiv \psi$, whenever $\mathbf{M}, k \models \varphi$ iff $\mathbf{M}, k \models \psi$ and $\mathbf{M}, k \mathrel{=\!\!|} \varphi$ iff $\mathbf{M}, k \mathrel{=\!\!|} \psi$, for any HT-trace $\mathbf{M}$ and any $k \in [0..\lambda)$. This last equivalence is the same as requiring that $\varphi \Leftrightarrow \psi$ is a tautology. Note that this relation, $\varphi \equiv \psi$, is stronger than coincidence of models $\mathrm{DHT}(\varphi) = \mathrm{DHT}(\psi)$. For instance, $\mathrm{DHT}(\bullet\top) = \mathrm{DHT}(\langle\tau^-\rangle\,\top) = \emptyset$ because models are checked at the initial situation $k = 0$ and there is no previous situation at that point, so $\mathrm{DHT}(\bullet\top) = \mathrm{DHT}(\perp)$. However, in general, $\bullet\top \not\equiv \perp$ since $\bullet\top$ is satisfied for any $k > 0$ (for instance $\circ\bullet\top \not\equiv \circ\perp$ but $\circ\bullet\top \equiv \top$ instead).

One interesting equivalence for explicit negation is

$$\sim\sim\varphi \equiv \varphi \quad (5)$$

since in HT or, any of its extensions, removing double default negation is not possible, that is, in general $\neg\neg\varphi \not\equiv \varphi$.

As with formulas, we say that path expressions ρ_1 and ρ_2 are *equivalent*, written $\rho_1 = \rho_2$, when $\|\rho_1\|^{\mathbf{M}}_R = \|\rho_2\|^{\mathbf{M}}_R$ for any HT-trace $\mathbf{M}$ and relation $R \in \{\models, \not\mathrel{=\!\!|}\}$. For instance, it is easy to see that:

$$\begin{array}{rclcrcl} (\rho_1;\rho_2);\rho_3 &=& \rho_1;(\rho_2;\rho_3) & \quad & \rho^* &=& \top? + (\rho;\rho^*) \\ \top?;\rho &=& \rho;\top? = \rho & & \rho;\rho^* &=& \rho^*;\rho \end{array}$$

[5] As we said before, following [11], we allow a propositional formula ϕ as an abbreviation of expression $(\phi?;\tau)$. That is, when no "?" is appended, we actually perform an additional one-step jump.

The following equivalences of path expressions proved in [7] allow us to push the converse operator inside, until it is only applied to $\top$. These equivalences are preserved with the introduction of explicit negation too:

Proposition 3 *For all path expressions ρ_1, ρ_2 and ρ and for all formulas φ, the following equivalences from [7] are maintained:*

$$(\rho^-)^- = \rho \qquad (\varphi?)^- = \varphi? \qquad (\rho^*)^- = (\rho^-)^*$$

$$(\rho_1 + \rho_2)^- = \rho_1^- + \rho_2^- \qquad (\rho_1;\rho_2)^- = \rho_2^-;\rho_1^-$$

Similarly, DHT with explicit negation also preserves a fundamental feature of HT called *persistence* as explained below.

Proposition 4 (Persistence) *For any* HT*-trace $\langle \mathbf{H}, \mathbf{T}\rangle$ of length λ, any dynamic formula φ and any path expression ρ, we have:*

1. $\langle \mathbf{H}, \mathbf{T}\rangle, k \models \varphi$ *implies* $\langle \mathbf{T}, \mathbf{T}\rangle, k \models \varphi$*, for all* $k \in [0..\lambda)$
2. $\langle \mathbf{H}, \mathbf{T}\rangle, k \mathrel{=\!\!|} \varphi$ *implies* $\langle \mathbf{T}, \mathbf{T}\rangle, k \mathrel{=\!\!|} \varphi$*, for all* $k \in [0..\lambda)$
3. $\|\rho\|^{\langle \mathbf{H},\mathbf{T}\rangle}_{\models} \subseteq \|\rho\|^{\langle \mathbf{T},\mathbf{T}\rangle}_{\models}$ *and* $\|\rho\|^{\langle \mathbf{H},\mathbf{T}\rangle}_{\not\mathrel{=\!\!|}} \supseteq \|\rho\|^{\langle \mathbf{T},\mathbf{T}\rangle}_{\not\mathrel{=\!\!|}}$. □

Persistence is a property known from intuitionistic logic; it expresses that accessible worlds satisfy the same or more formulas than the current world, where $\mathbf{T}$ is "accessible" from $\mathbf{H}$ in HT. This also explains the semantics of $[\rho]\,\varphi$, which behaves as a kind of intuitionistic implication (used to define '$\rightarrow$' as a derived operator) and so, it must hold for all accessible worlds, viz. $\langle \mathbf{H}, \mathbf{T}\rangle$ and $\langle \mathbf{T}, \mathbf{T}\rangle$.

As explained in [7], DHT collapses to LDL when we force total traces $\langle \mathbf{T}, \mathbf{T}\rangle$, so we write $\mathbf{T}, k \models \varphi$ to represent LDL satisfaction, which in fact can be just seen as an abbreviation of $\langle \mathbf{T}, \mathbf{T}\rangle, k \models \varphi$. Similarly, the accessibility relation for path expressions in LDL corresponds here to $\|\rho\|^{\langle \mathbf{T},\mathbf{T}\rangle}_{\models}$ or just $\|\rho\|^{\mathbf{T}}_{\models}$. Since our new Definition 1 is also equipped with falsification and explicit negation, it provides an explicit negation operator for LDL as a side effect too: it just amounts to restrict traces to be total. By doing so, the main relevant change occurs in the interpretation of $[\cdot]$ which, as expected, becomes simpler. Conditions 4 and 9 become:

4'. $\mathbf{M}, k \models [\rho]\,\varphi$ if $\mathbf{M}, i \models \varphi$ for all i, $(k, i) \in \|\rho\|^{\mathbf{M}}_{\models}$

9'. $\mathbf{M}, k \mathrel{=\!\!|} [\rho]\,\varphi$ if $\mathbf{M}, i \mathrel{=\!\!|} \varphi$ for some i, $(k, i) \in \|\rho\|^{\mathbf{M}}_{\models}$

where $\mathbf{M} = \langle \mathbf{T}, \mathbf{T}\rangle$. The reader may wonder why relation $\|\rho\|^{\mathbf{M}}_{\models}$ is preserved in both items 4' and 9' rather than switched as happens with $\models$ versus $\mathrel{=\!\!|}$. The reason is that $[\rho]\,\varphi$ is a kind of implication that, to become false, must make its antecedent about (k, i) true and make its consequent about φ false. Total interpretations (i.e., LDL interpretations) can be forced by the inclusion of the excluded middle axiom for default negation:

$$[\tau^*]\,(L \vee \neg L) \quad (6)$$

for any explicit literal L of the form p or $\sim p$ and any atom $p \in \mathcal{A}$.

The next theorem shows the derived satisfaction and falsification relations for temporal operators from LTL. As intended, satisfaction coincides with the one from standard THT with finite traces [9], but we also provide explicit negation and falsification for this logic, something not achieved so far.

Theorem 2 *Let $\mathbf{M} = \langle \mathbf{H}, \mathbf{T}\rangle$ be an* HT*-trace of length λ over alphabet $\mathcal{A}$ and $k \in [0..\lambda)$. Given the respective definitions of derived operators, we get the following satisfaction and falsification conditions:*

1. $\mathbf{M}, k \models \mathsf{F}$ *iff* $k + 1 = \lambda$

2. $\mathbf{M}, k \Dashv \mathbf{F}$ *iff* $k+1 < \lambda$
3. $\mathbf{M}, k \models \circ\alpha$ *iff* $k+1 < \lambda$ *and* $\mathbf{M}, k+1 \models \alpha$
4. $\mathbf{M}, k \Dashv \circ\alpha$ *iff* $k+1 = \lambda$ *or* $\mathbf{M}, k+1 \Dashv \alpha$
5. $\mathbf{M}, k \models \widehat{\circ}\alpha$ *iff* $k+1 = \lambda$ *or* $\mathbf{M}, k+1 \models \alpha$
6. $\mathbf{M}, k \Dashv \widehat{\circ}\alpha$ *iff* $k+1 < \lambda$ *and* $\mathbf{M}, k+1 \Dashv \alpha$
7. $\mathbf{M}, k \models \Diamond\alpha$ *iff* $\mathbf{M}, i \models \alpha$ *for some* $i \in [k..\lambda)$
8. $\mathbf{M}, k \Dashv \Diamond\alpha$ *iff* $\mathbf{M}, i \Dashv \alpha$ *for all* $i \in [k..\lambda)$
9. $\mathbf{M}, k \models \Box\alpha$ *iff* $\mathbf{M}, i \models \alpha$ *for all* $i \in [k..\lambda)$
10. $\mathbf{M}, k \Dashv \Box\alpha$ *iff* $\mathbf{M}, i \Dashv \alpha$ *for some* $i \in [k..\lambda)$
11. $\mathbf{M}, k \models \alpha \,\mathbf{U}\, \beta$ *iff for some* $i \in [k..\lambda)$*, we have* $\mathbf{M}, i \models \beta$ *and* $\mathbf{M}, j \models \alpha$ *for all* $j \in [k..i)$
12. $\mathbf{M}, k \Dashv \alpha \,\mathbf{U}\, \beta$ *iff for all* $i \in [k..\lambda)$*, we have* $\mathbf{M}, i \Dashv \beta$ *or* $\mathbf{M}, j \Dashv \alpha$ *for some* $j \in [k..i)$

The proof of the previous theorem follows more or less directly from the definitions, but proving the derived semantics for the release operator is more involved. We provide the characterisation of release in the following two propositions.

Proposition 5 *Given two formulas* α *and* β *and an HT-interpretation* $\mathbf{M}$*, the following assertions are equivalent:*

1. $\mathbf{M}, k \models \beta \,\mathbf{U}\, (\alpha \wedge \beta) \vee \Box\beta$ *(or* $\mathbf{M}, k \models \alpha \,\mathbf{R}\, \beta$*)*
2. *for all* $i \in [k..\lambda)$*, we have that* $\mathbf{M}, i \models \beta$ *or* $\mathbf{M}, j \models \alpha$*, for some* $j \in [k..i)$ *(or* $\mathbf{M}, k \Dashv\ \sim\alpha \,\mathbf{U} \sim\beta$*)*

Proof . Suppose that 1 is true. If $\mathbf{M}, k \models \Box\beta$, then we deduce 2. If $\mathbf{M}, k \not\models \Box\beta$, take:

$$i \stackrel{def}{=} min\{j \geq k \mid \mathbf{M}, j \not\models \beta\}$$

Then $\mathbf{M}, i \not\models \beta$ and $\mathbf{M}, j \models \beta$ for any $k \leq j < i$. On the other hand, $\mathbf{M}, k \models \beta \,\mathbf{U}\, (\alpha \wedge \beta)$, so there exists $s \geq k$ such that $\mathbf{M}, s \models \alpha$, $\mathbf{M}, s \models \beta$ and $\mathbf{M}, j \models \beta$ for any $k \leq j < s$. Take $r \geq k$ and suppose that $M, r \not\models \beta$. We would like to show that $\mathbf{M}, j \models \alpha$ for some $k \leq j < r$. By definition of i, we can say that $r \leq i$. If $\mathbf{M}, j \not\models \alpha$ for all $k \leq j < r$, then $s \geq r$ (since $\mathbf{M}, s \models \alpha$) and $s > r$ (since $\mathbf{M}, s \models \beta$ and $\mathbf{M}, r \not\models \beta$). But then $M, r \not\models \beta$ and $r < s$ which contradicts the fact that $\mathbf{M}, j \models \beta$ for any $k \leq j < s$.

Now suppose that 2 is true. If $\mathbf{M}, k \models \Box\beta$, then we obviously deduce 1. If $\mathbf{M}, k \not\models \Box\beta$, take:

$$i \stackrel{def}{=} min\{j \geq k \mid \mathbf{M}, j \not\models \beta\}$$

Since $\mathbf{M}, i \not\models \beta$, by 2) we know that $\mathbf{M}, j \models \alpha$ for some $k \leq j < i$. This implies that $\mathbf{M}, j \models \beta$. Finally, if $k \leq r < j$, then $r < i$, so $\mathbf{M}, r \models \beta$. We conclude that $\mathbf{M}, k \models \beta \,\mathbf{U}\, (\alpha \wedge \beta)$. □

Proposition 6 *Given two formulas* α *and* β *and an HT-interpretation* $\mathbf{M}$*, the following assertions are equivalent:*

1. $\mathbf{M}, k \Dashv \beta \,\mathbf{U}\, (\alpha \wedge \beta) \vee \Box\beta$ *(or* $\mathbf{M}, k \Dashv \alpha \,\mathbf{R}\, \beta$*)*
2. *for some* $i \in [k..\lambda)$*, we have that* $\mathbf{M}, i \Dashv \beta$ *and* $\mathbf{M}, j \Dashv \alpha$*, for any* $j \in [k..i)$ *(or* $\mathbf{M}, k \models \sim\alpha \,\mathbf{U} \sim\beta$*)*

Proof . Suppose that 1 is true, then $\mathbf{M}, k \Dashv \Box\beta$ and $\mathbf{M}, k \Dashv \beta \,\mathbf{U}\, (\alpha \wedge \beta)$. Take:

$$i \stackrel{def}{=} min\{j \geq k \mid \mathbf{M}, j \Dashv \beta\}$$

Then $\mathbf{M}, i \Dashv \beta$ and $\mathbf{M}, j \not\Dashv \beta$ for any $k \leq j < i$. If $\mathbf{M}, r \not\Dashv \alpha$, for some $k \leq r < i$, then $(k, r) \in \|(\beta?; \tau)^*\|^{\mathbf{M}}_{\not\Dashv}$ because if $k \leq j < r$, then $j < i$ and $\mathbf{M}, j \not\Dashv \beta$. Then $\mathbf{M}, r \Dashv \alpha$ or $\mathbf{M}, r \Dashv \beta$. Since $r < i$, $\mathbf{M}, r \not\Dashv \beta$ so we would have that $\mathbf{M}, r \Dashv \alpha$ which is a contradiction.

Now suppose that 2 is true. Then $\mathbf{M}, k \Dashv \Box\beta$. Take:

$$i \stackrel{def}{=} min\{j \geq k \mid \mathbf{M}, j \Dashv \beta\}$$

Then $\mathbf{M}, i \Dashv \beta$ and $\mathbf{M}, j \not\Dashv \beta$ for any $j < i$. Suppose that $(k, r) \in \|(\beta?; \tau)^*\|^{\mathbf{M}}_{\not\Dashv}$, this implies that $\mathbf{M}, j \not\Dashv \beta$ for any $k \leq j < r$, so $i \geq r$ because $\mathbf{M}, i \Dashv \beta$. If $i = r$, then $\mathbf{M}, r \Dashv \beta$ and, if $r < i$, then $\mathbf{M}, r \Dashv \alpha$. Any case, we have show that $\mathbf{M}, r \Dashv \alpha \wedge \beta$ which shows that $\mathbf{M}, k \Dashv \beta \,\mathbf{U}\, (\alpha \wedge \beta)$. □

These two propositions allow us to prove an interesting De Morgan-style relation (through explicit negation) between until and release:

Theorem 3 *For any pair of formulas* α *and* β*, we have*

$$\alpha \,\mathbf{U}\, \beta \quad \equiv \quad \sim(\sim\alpha \,\mathbf{R} \sim\beta) \tag{7}$$

$$\alpha \,\mathbf{R}\, \beta \quad \equiv \quad \sim(\sim\alpha \,\mathbf{U} \sim\beta) \tag{8}$$

Proof . For proving (7), we have to show that, for any HT-interpretation $\mathbf{M}$ and $k \in [0..\lambda)$:

$$\mathbf{M}, k \models \alpha \,\mathbf{U}\, \beta \text{ iff } \mathbf{M}, k \Dashv\ \sim\alpha \,\mathbf{R} \sim\beta$$

and

$$\mathbf{M}, k \Dashv \alpha \,\mathbf{U}\, \beta \text{ iff } \mathbf{M}, k \models \sim\alpha \,\mathbf{R} \sim\beta$$

By Proposition 6, $\mathbf{M}, k \Dashv\ \sim\alpha \,\mathbf{R} \sim\beta$ iff $\mathbf{M}, k \models \alpha \,\mathbf{U}\, \beta$. On the other hand, Proposition 5 implies that $\mathbf{M}, k \models \sim\alpha \,\mathbf{R} \sim\beta$ iff $\mathbf{M}, k \Dashv \alpha \,\mathbf{U}\, \beta$.

For proving (8) it suffices to see:

$$\begin{array}{rcll} \sim(\sim\alpha \,\mathbf{U} \sim\beta) & \equiv & \sim(\sim(\sim\sim\alpha \,\mathbf{R} \sim\sim\beta)) & \text{by (7)} \\ & \equiv & \alpha \,\mathbf{R}\, \beta & \text{by (5)} \end{array}$$

□

This result is interesting because, in regular THT, these De Morgan-style equivalences do not hold for default negation. Moreover, [4] proved that $\mathbf{R}$ cannot be defined in terms of $\mathbf{U}$ and $\circ$ in standard THT. Theorem 3 is showing that, once explicit negation is introduced, we can indeed define release in terms of until and vice versa.

It is perhaps worth to wonder whether $[\cdot]$ and $\langle\cdot\rangle$ are also inter-definable using a similar De Morgan-style analogy, say $[\rho]\,\varphi \stackrel{?}{\equiv} \sim\langle\rho\rangle \sim\varphi$. Unfortunately, this does not hold since necessity involves a kind of implication, possibility a conjunction, and the former cannot be represented in terms of the latter in HT, even by introducing explicit negation. In the general case, we can prove, at most, one of the directions for De Morgan-style properties. However, if ρ is test-free, then both directions are guaranteed. This is formally stated below:

Theorem 4 *Given a regular expression* ρ*, two formulas* φ *and* ψ *and an HT-trace* $\mathbf{M}$*, we always have:*

1. $\mathbf{M}, k \models \sim\langle\rho\rangle \sim\varphi$ *implies* $\mathbf{M}, k \models [\rho]\,\varphi$ *for any* $k \in [0..\lambda)$
2. $\mathbf{M}, k \Dashv \sim\langle\rho\rangle \sim\varphi$ *implies* $\mathbf{M}, k \Dashv [\rho]\,\varphi$ *for any* $k \in [0..\lambda)$
3. *If* ρ *is test-free, then:*

$$\sim\langle\rho\rangle \sim\varphi \equiv [\rho]\,\varphi \text{ and } \sim[\rho] \sim\varphi \equiv \langle\rho\rangle\,\varphi$$

Proof . First, notice that, if $\mathbf{M}, k \Dashv \langle\rho\rangle \sim\varphi$ and $(k, i) \in \|\rho\|^{\mathbf{M}}_{\models}$, then $(k, i) \in \|\rho\|^{\mathbf{M}}_{\not\Dashv}$ by Proposition 2, so $\mathbf{M}, i \models \varphi$. Moreover, when $(k, i) \in \|\rho\|^{\mathbf{T}}_{\models}$, we can apply Proposition 4 and Proposition 2 to deduce that $(k, i) \in \|\rho\|^{\mathbf{M}}_{\not\Dashv}$, so $\mathbf{M}, i \models \varphi$ and $\mathbf{T}, i \models \varphi$.

On the other hand, if $\mathbf{M}, k \models\!\!\!\!\dashv \ \sim\langle\rho\rangle \sim\varphi$ or, equivalently, $\mathbf{M}, k \models \langle\rho\rangle \sim\varphi$, we know that there exists $(k,i) \in \|\rho\|^{\mathbf{M}}_{\models}$ such that $\mathbf{M}, i =\!\!\mid \varphi$. Since $\|\rho\|^{\mathbf{M}}_{\models} \subseteq \|\rho\|^{\mathbf{T}}_{\models}$, we deduce that $\mathbf{M}, k =\!\!\mid [\rho]\,\varphi$.
The equivalence between the formulas when ρ is test-free follows from Proposition 2. □

Notice that, when ρ is not test-free we can not guarantee this result anymore. In fact, take α and β two formulas. Then:

$$\sim\langle\alpha?\rangle \sim\beta \equiv \sim(\alpha \wedge \sim\beta) \equiv \sim\alpha \vee \beta,$$

whereas:

$$[\alpha?]\,\beta \equiv \alpha \rightarrow \beta$$

and $\sim\alpha \vee \beta$ and $\alpha \rightarrow \beta$ are not equivalent in HT with explicit negation.

On the other hand,

$$\sim[\alpha?] \sim\beta \equiv \sim(\alpha \rightarrow \sim\beta) \equiv_w (\neg\neg\alpha \wedge \beta),$$

and:

$$\langle\alpha?\rangle\,\beta \equiv \alpha \wedge \beta.$$

Theorem 4 can be used on temporal operators defined with test-free path expressions, like $\Diamond, \Box, \circ$ and $\widehat{\circ}$, so we immediately obtain:

Corollary 1 *Given a formula φ, we always have:*

- $\circ\varphi \equiv \sim\widehat{\circ} \sim\varphi$
- $\widehat{\circ}\varphi \equiv \sim\circ \sim\varphi$
- $\Diamond\varphi \equiv \sim\Box \sim\varphi$
- $\Box\varphi \equiv \sim\Diamond \sim\varphi$

Again, this result is interesting because for standard THT, [4] also proved that $\Box$ cannot be defined in terms of $\Diamond$ and $\circ$, while this can be done now by resorting to explicit negation. Another interesting application of this result is that it provides a new method for proving THT equivalences by resorting to De-Morgan-style transformations using $\sim$ for the proofs. For instance, suppose we have developed a proof for the THT tautology:

$$\Diamond p \Leftrightarrow p \vee \circ\Diamond p \tag{9}$$

The dual property $\Box p \Leftrightarrow p \wedge \widehat{\circ}\Box p$ cannot be directly derived from (9) in THT as we would do in LTL by introducing negations $\neg$, but can be now derived if we use the explicit negation $\sim$ instead. In this way, we can construct the following sequence of equivalent formulas:

$\Diamond\sim p$	$\Leftrightarrow$	$\sim p \vee \circ\Diamond\sim p$	Replacing p by $\sim p$ in (9)
$\sim\Diamond\sim p$	$\Leftrightarrow$	$\sim(\sim p \vee \circ\Diamond\sim p)$	Property of $\Leftrightarrow$
$\sim\Diamond\sim p$	$\Leftrightarrow$	$\sim\sim p \wedge \sim\circ\Diamond\sim p$	De Morgan on $\vee$
$\Box p$	$\Leftrightarrow$	$p \wedge \widehat{\circ}\sim\Diamond\sim p$	Corollary 1; double negation
$\Box p$	$\Leftrightarrow$	$p \wedge \widehat{\circ}\Box p$	Corollary 1

We conclude this section by observing that, in the same way that DHT collapses to LDL by adding axiom (6) (i.e. forcing total traces), when we restrict the syntax to temporal theories, THT collapses to LTL. Therefore, Definition 1 is also providing a semantics for LTL with explicit negation.

3 The *While* operator

In the previous section, we introduced a new temporal operator $\varphi \mathbin{\mathbf{W}} \psi$ defined as $[(\psi?;\tau)^*]\,\varphi$ and read as "repeat φ while ψ." This operator is not usual in LTL because, in that logic, it can be easily defined in terms of $\mathbf{U}$ or $\mathbf{R}$:

Proposition 7 *In LTL, we have the following equivalences:*

$$\begin{aligned} \alpha \mathbin{\mathbf{W}} \beta &\equiv \neg(\beta \mathbin{\mathbf{U}} \neg\alpha) \equiv \neg\beta \mathbin{\mathbf{R}} \alpha \\ \alpha \mathbin{\mathbf{U}} \beta &\equiv \neg\neg(\alpha \mathbin{\mathbf{U}} \neg\neg\beta) \\ &\equiv \neg(\neg\beta \mathbin{\mathbf{W}} \alpha) \end{aligned}$$ □

In THT, however, these equivalences do not hold. Given $\mathbf{M} = \langle\mathbf{H},\mathbf{T}\rangle$, the derived semantics that we obtain for $\mathbf{W}$ is:

- $\mathbf{M}, k \models \varphi \mathbin{\mathbf{W}} \psi$ iff for all $i \in [k..\lambda)$ such that for all $j \in [k..i)$ $\mathbf{M}', j \models \psi$ implies $\mathbf{M}', i \models \varphi$, for both $\mathbf{M}' = \mathbf{M}$ and $\mathbf{M}' = \langle\mathbf{T},\mathbf{T}\rangle$;
- $\mathbf{M}, k =\!\!\mid \varphi \mathbin{\mathbf{W}} \psi$ iff for some $i \in [k..\lambda)$ we have $\mathbf{M}, i =\!\!\mid \varphi$ and for all $j \in [k..i)$ $\langle\mathbf{T},\mathbf{T}\rangle, j \models \psi$.

This means that, unlike $\mathbf{U}$ and $\mathbf{R}$, operator $\mathbf{W}$ has an implicational nature. As a consequence, there is no clear way to represent $\mathbf{W}$ in terms of the other operators[6]. For instance, for the connection between $\mathbf{U}$ and $\mathbf{W}$ through explicit negation we can just guarantee:

Theorem 5 *Given two formulas φ and ψ, an HT-trace $\mathbf{M} = \langle\mathbf{H},\mathbf{T}\rangle$ and any $k \in [0..\lambda)$, the following conditions hold*

1. $\mathbf{M}, k \models \sim(\varphi \mathbin{\mathbf{U}} \psi)$ *implies* $\mathbf{M}, k \models \sim\psi \mathbin{\mathbf{W}} \varphi$
2. $\mathbf{M}, k =\!\!\mid \sim(\varphi \mathbin{\mathbf{U}} \psi)$ *implies* $\mathbf{M}, k =\!\!\mid \sim\psi \mathbin{\mathbf{W}} \varphi$
3. $\mathbf{M}, k \models \sim(\varphi \mathbin{\mathbf{W}} \psi)$ *iff* $\mathbf{M}, k \models \neg\neg\psi \mathbin{\mathbf{U}} \sim\varphi$
4. $\mathbf{M}, k =\!\!\mid \neg\neg\psi \mathbin{\mathbf{U}} \sim\varphi$ *implies* $\mathbf{M}, k =\!\!\mid \sim(\varphi \mathbin{\mathbf{W}} \psi)$

Proof . The first and second assertions can be proved by applying Theorem 4, because $\varphi \mathbin{\mathbf{U}} \psi = \langle(\varphi?;\tau)^*\rangle\,\psi$ and $\sim\psi \mathbin{\mathbf{W}} \varphi = [(\varphi?;\tau)^*] \sim\psi$.
On the other hand, $\mathbf{M}, k \models \sim (\varphi \mathbin{\mathbf{W}} \psi)$ iff there exists $(k,i) \in \|(\psi?;\tau)^*\|^{\mathbf{T}}_{\models}$ and $\mathbf{M}, i =\!\!\mid \varphi$. Since $\|(\psi?;\tau)^*\|^{\mathbf{T}}_{\models} = \|(\neg\neg\psi?;\tau)^*\|^{\mathbf{M}}_{\models}$, this is equivalent to say that $\mathbf{M}, k \models \neg\neg\psi \mathbf{U} \sim\varphi$.
For the last point, suppose that $\mathbf{M}, k =\!\!\mid \neg\neg\psi \mathbf{U} \sim\varphi$. Then, for any $(k,i) \in \|(\neg\neg\psi?;\tau)^*\|^{\mathbf{M}}_{\not=\!\mid}$, we have that $\mathbf{M}, i =\!\!\mid \ \sim\varphi$ or $\mathbf{M}, i \models \varphi$. This implies that $\mathbf{M}, k \models \varphi \mathbin{\mathbf{W}} \psi$ because:

$$\|(\psi?;\tau)^*\|^{\mathbf{M}}_{\models} \subseteq \|(\psi?;\tau)^*\|^{\mathbf{T}}_{\models}$$

and

$$\|(\psi?;\tau)^*\|^{\mathbf{T}}_{\models} = \|(\neg\neg\psi?;\tau)^*\|^{\mathbf{M}}_{\models} \subseteq \|(\neg\neg\psi?;\tau)^*\|^{\mathbf{M}}_{\not=\!\mid}$$

In general, the other implication is not true. Take $\mathbf{M}$ defined by $H_0 = \{p\} \subseteq T_0 = \{p,q\}$, $H_1 = \emptyset \subseteq T_1 = \{p,q\}$ and $H_i = \emptyset \subseteq T_i = \{p\}$, for any $i \geq 2$. Then $\mathbf{M}, 0 \models p \mathbin{\mathbf{W}} q$ since $(0,i) \notin \|(q?;\tau)^*\|^{\mathbf{M}}_{\models}$ for $i \geq 1$ ($q \notin H_0$) and, if $(0,i) \in \|(q?;\tau)^*\|^{\mathbf{T}}_{\models}$, then $i \leq 2$ ($q \notin T_j$ for $j \geq 2$) and $\mathbf{T}, i \models p$ for $i = 0,1,2$. On the other hand, $\mathbf{M}, 0 \not=\!\mid \neg\neg q \mathbf{U} \sim p$ because $\mathbf{M}, 1 \not=\!\mid \ \sim p$ ($\mathbf{M}, 1 \not\models p$) and $\mathbf{M}, 0 \not=\!\mid \neg\neg q$ ($\mathbf{M}, 0 \not=\!\mid \neg\neg q$ iff $\mathbf{M}, 0 \not\models \sim\neg\neg q$ iff $\mathbf{M}, 0 \not\models \neg\neg\neg q$ iff $\mathbf{M}, 0 \not\models \neg q$ iff $q \in T_0$).[7] □

The following equivalences describe these three operators, $\mathbf{U}$, $\mathbf{R}$ and $\mathbf{W}$, by an inductive unfolding.

Proposition 8 *In THT we have the following equivalences:*

$$\begin{aligned} \alpha \mathbin{\mathbf{U}} \beta &\equiv \beta \vee (\alpha \wedge \circ(\alpha \mathbin{\mathbf{U}} \beta)) \\ \alpha \mathbin{\mathbf{R}} \beta &\equiv \beta \wedge (\alpha \vee \circ(\alpha \mathbin{\mathbf{R}} \beta)) \\ \alpha \mathbin{\mathbf{W}} \beta &\equiv \alpha \wedge (\beta \rightarrow \circ(\alpha \mathbin{\mathbf{W}} \beta)) \end{aligned}$$ □

[6] We conjecture that this may be, in fact, a fundamental operator in THT, even without explicit negation.
[7] For any formula φ, $\mathbf{M}, k \models \sim\neg\varphi$ iff $\mathbf{M}, k \models \neg\neg\varphi$

The first two equivalences are well-known from LTL and also preserved in THT. The third equivalence shows that the $\mathbf{W}$ operator is formed by a repeated application of implications. This is even clearer in the following expansion:

Proposition 9 *The formula $\alpha \mathbf{W} \beta$ is THT equivalent to the (possibly infinite) conjunction of rules:*

$$\left(\bigwedge_{j=0}^{i-1} \circ^j \beta \right) \quad \rightarrow \quad \circ^i \alpha$$

for all $i \geq 0$. □

To see how this expression works, think about the formula $w \ \mathbf{W} \ f$ where w means "pouring water" and f means "fire." Notice that, for $i = 0$, the rule body becomes an empty conjunction ($\top$) and so this expands to $\top \rightarrow \circ^0 w$ which is just equivalent to fact w. For instance, for $i \in [0, 3]$ we get the rules:

$$\begin{array}{rcl} & & w \\ f & \rightarrow & \circ w \\ f \wedge \circ f & \rightarrow & \circ^2 w \\ f \wedge \circ f \wedge \circ^2 f & \rightarrow & \circ^3 w \\ & \vdots & \end{array}$$

That is, we start pouring water at situation 0 and, if we have fire, we keep pouring water at 1 and check fire again, and so on. As we can see, the effect of each f test is placed at the next situation. Thus, the reading of $w \ \mathbf{W} \ f$ is like a procedural program "**do** pour-water **while** fire". If we want to move the test to the same situation of its effect, we would write instead $(f \rightarrow w) \ \mathbf{W} \ f$ whose reading would be ''**while** fire **do** pour-water" and whose expansion becomes:

$$\begin{array}{rcl} f & \rightarrow & w \\ f \wedge \circ f & \rightarrow & \circ w \\ f \wedge \circ f \wedge \circ^2 f & \rightarrow & \circ^2 w \\ & \vdots & \end{array}$$

The expansion of $(w \mathbf{W} f)$ as a set of rules reveals its behaviour from a logic programming point of view. For instance, a theory only containing the formula $(w \mathbf{W} f)$ has a unique temporal stable model (per each length $\lambda > 0$) in which w is only true at the initial state, whereas f is always false, since there is no additional evidence about fire. In LTL, the formula $(w \mathbf{W} f)$ is equivalent to $(\neg f \mathbf{R} w)$, that is, $\Box w \vee (w \mathbf{U} (w \wedge \neg f))$. In our formalism, however, this last formula produces temporal stable models with arbitrary prefix sequences of w, and even the case in which w holds in all the states of the trace. In other words, water may be poured arbitrarily many times, even though fire is false all over the trace.

4 Conclusions

We have introduced the explicit negation operator in Linear Dynamic Equilibrium Logic (DEL) and its monotonic basis, Linear Dynamic Here-and-There (DHT). The introduction of explicit negation was done by adding a second relation of "falsification" dual to the standard satisfaction. As a result, we obtain not only a characterisation of explicit negation as a regular logical connective that can be combined with modal operators, but also interesting results showing that some temporal operators become inter-definable, when this was not possible without explicit negation. We have also discovered and explained a new temporal operator we called "while," that is an implicational dual of until and may be useful as a basic connective for temporal logic programming. We conjecture that this new while operator cannot be reduced to the other temporal connectives, even with the possible use of explicit negation. We leave the proof of this conjecture for future work. We also plan to add explicit negation to first order temporal and dynamic theories and study its consequences for temporal logic programs with variables [3].

ACKNOWLEDGEMENTS

We are grateful to Torsten Schaub and David Pearce for their comments and suggestions, and to the anonymous reviewers whose work has helped us to improve the paper. This work was partially supported by Ministry of Science and Innovation, Spain (grant TIC2017-84453-P) and Xunta de Galicia, Spain (grants GPC ED431B 2019/03 and 2016-2019 ED431G/01, CITIC).

REFERENCES

[1] Felicidad Aguado, Pedro Cabalar, Martín Diéguez, Gilberto Pérez, and Concepción Vidal, 'Temporal equilibrium logic: a survey', *Journal of Applied Non-Classical Logics*, **23**(1-2), 2–24, (2013).

[2] Felicidad Aguado, Pedro Cabalar, Jorge Fandinno, David Pearce, Gilberto Pérez, and Concepción Vidal, 'Revisiting explicit negation in answer set programming', *Theory and Practice of Logic Programming*, **19**(5-6), 908—924, (2019).

[3] Felicidad Aguado, Pedro Cabalar, Gilberto Pérez, Concepción Vidal, and Martín Diéguez, 'Temporal logic programs with variables', *Theory and Practice of Logic Programming*, **17**(2), 226–243, (2017).

[4] Philippe Balbiani, Joseph Boudou, Martín Diéguez, and David Fernández-Duque, 'Bisimulations for intuitionistic temporal logics', *CoRR*, **abs/1803.05078**, (2018).

[5] Anne-Gwenn Bosser, Pedro Cabalar, Martín Diéguez, and Torsten Schaub, 'Introducing temporal stable models for linear dynamic logic', in *Proceedings of the 16th International Conference on Principles of Knowledge Representation and Reasoning, (KR 2018)*, eds., Michael Thielscher, Francesca Toni, and Frank Wolter, pp. 12–21, (2018).

[6] Tom Bylander, 'The computational complexity of propositional strips planning', *Artificial Intelligence*, **69**(1), 165 – 204, (1994).

[7] Pedro Cabalar, Martín Diéguez, and Torsten Schaub, 'Towards dynamic answer set programming over finite traces', in *Proceedings of the 15th International Conference on Logic Programming and Nonmonotonic Reasoning,(LPNMR 2019)*, eds., Marcello Balduccini, Yuliya Lierler, and Stefan Woltran, volume 11481 of *Lecture Notes in Computer Science*, pp. 148–162. Springer, (2019).

[8] Pedro Cabalar, Roland Kaminski, Philip Morkisch, and Torsten Schaub, 'telingo = ASP + time', in *Proceedings of the 15th International Conference on Logic Programming and Nonmonotonic Reasoning, (LPNMR 2019)*, eds., Marcello Balduccini, Yuliya Lierler, and Stefan Woltran, pp. 256–269, (2019).

[9] Pedro Cabalar, Roland Kaminski, Torsten Schaub, and Anna Schuhmann, 'Temporal answer set programming on finite traces', *Theory and Practice of Logic Programming*, **18**(3-4), 406–420, (2018).

[10] Pedro Cabalar and Gilberto Peréz, 'Temporal Equilibrium Logic: A First Approach', in *Proceedings of the 11th International Conference on Computer Aided Systems Theory (EUROCAST'07)*, p. 241–248, (2007).

[11] Giuseppe De Giacomo and Moshe Y. Vardi, 'Linear temporal logic and linear dynamic logic on finite traces', in *Proceedings of the 23rd International Joint Conference on Artificial Intelligence (IJCAI 2013)*, ed., Francesca Rossi, pp. 854–860, (2013).

[12] Michael Gelfond and Vladimir Lifschitz, 'The stable models semantics for logic programming', in *Proceedings of the 5th International Conference on Logic Programming*, pp. 1070–1080, (1988).

[13] Michael Gelfond and Vladimir Lifschitz, 'Classical negation in logic programs and disjunctive databases', *New Generation Computing*, **9**(3/4), 365–386, (1991).

[14] Michael Gelfond and Vladimir Lifschitz, 'Representing action and change by logic programs', *Journal of Logic Programming*, **17**(2/3&4), 301–321, (1993).

[15] David Harel, Dexter Kozen, and Jerzy Tiuryn, *Dynamic Logic*, MIT Press, 2000.

[16] Arend Heyting, 'Die formalen Regeln der intuitionistischen Logik', in *Sitzungsberichte der Preussischen Akademie der Wissenschaften*, 42–56, Deutsche Akademie der Wissenschaften zu Berlin, (1930). Reprint in Logik-Texte: Kommentierte Auswahl zur Geschichte der Modernen Logik, Akademie-Verlag, 1986.

[17] Victor Marek and Miroslaw Truszczyński, *Stable models and an alternative logic programming paradigm*, 169–181, Springer-Verlag, 1999.

[18] Ilkka Niemelä, 'Logic programs with stable model semantics as a constraint programming paradigm', *Annals of Mathematics and Artificial Intelligence*, **25**, 241–273, (1999).

[19] David Pearce, 'A New Logical Characterisation of Stable Models and Answer Sets', in *Proceedings of Non-Monotonic Extensions of Logic Programming (NMELP'96)*, pp. 57–70, Bad Honnef, Germany, (1996).

[20] Amir Pnueli, 'The temporal logic of programs', in *Proceedings of the 18th Annual Symposium on Foundations of Computer Science*, SFCS '77, pp. 46–57, Washington, DC, USA, (1977). IEEE Computer Society.