Dynamic Equilibrium Logic for Action and Change: Abridged report

Arvid Becker[1,3], Martín Diéguez[2], Susana Hahn[1,3] and Torsten Schaub[1,3]

[1]University of Potsdam, Germany
[2]University of Angers, France
[3]Potassco Solutions GmbH, Germany

Abstract

Reasoning about action and change is a fundamental challenge in Artificial Intelligence, yet dedicated action languages within Answer Set Programming (ASP) see significantly less adoption than plain ASP. In practice, the lack of a unified approach inhibits the creation of an expressive, sustainable framework that leverages ASP's rich modeling capabilities and efficient solvers. To bridge this gap, we introduce a non-monotonic modeling framework based on Dynamic Logic (DL) that is designed for direct implementation through modern ASP systems. By synthesizing and adapting variants of established methodologies, our approach, the Dynamic Equilibrium Logic with Actions and Fluents (DEL^{af}), unifies a rich two-sorted language with non-monotonic semantic foundations. As common in DL, it explicitly distinguishes between state variables (fluents) and action variables to facilitate intuitive knowledge representation, combining propositional logic for concurrent state transitions with DL-based path expressions for global constraints over complex trajectories. Furthermore, we demonstrate that despite its expressive two-sorted nature, our formalism can be entirely reduced to the simpler, one-sorted Dynamic Equilibrium Logic (DEL). This strict reduction paves the way for practical implementation, ultimately allowing these expressive dynamic problems to be seamlessly solved using the temporal ASP solver *telingo*.

CEUR-WS.org/Vol-4311/ASPOCP_regular_03.pdf

1. Introduction

Reasoning about action and change is a fundamental challenge in Artificial Intelligence. While various action languages [1, 2] and temporal extensions [3, 4, 5] have been proposed within the context of Answer Set Programming (ASP; [6]), their adoption significantly trails that of plain ASP. In practice, it appears that ASP's pragmatic advantages, namely, its rich modeling language and efficient solvers, often outweigh the specific benefits of these dedicated approaches. Furthermore, the lack of a unified approach inhibits the creation of an expressive, sustainable framework for dealing with dynamic systems in ASP. To consolidate these directions, we introduce a rich modeling language for reasoning about action and change, designed for direct implementation through ASP solvers.

To this end, we adopt Dynamic Logic (DL; [7]) as our formal foundation, combining the language of [8, 9] with the semantic framework of [10, 11]. Although originally conceived for reasoning about programs, DL is equally well-suited for reasoning about action and change [12]. Standard DL and its early derivatives for reasoning about action and change [8, 9, 13] are inherently monotonic and rely on branching time. Furthermore, they strictly separate state variables (fluents) from action variables. State variables capture static properties evaluated within individual states, whereas action variables represent dynamic events that induce transitions between these states. These similarities carry over to the DL-inspired formalism of *golog* [14]. In this light, our approach can be viewed as a linear-time, non-monotonic extension of [8, 9]. Rebasing the semantics of DL on linear-time structures was advocated in [15] and recently gained traction through the work of [16]. Although syntactically restricted, the linear dynamic logic of [15] provides the foundation for the approach of [17] to reasoning about action and change; it models inertia using successor-state-like axioms in a monotonic setting. Building on this, [4] introduced a genuinely non-monotonic variant by extending the traditional reduct-based semantics

Joint Proceedings of the Workshops and Doctoral Consortium of the 42nd International Conference on Logic Programming (ICLP 2026) co-located with the Federated Logic Conference (FLoC 2026).

arvid.becker@uni-potsdam.de (A. Becker); martin.dieguezlodeiro@univ-angers.fr (M. Diéguez); hahnmartinlu@uni-potsdam.de (S. Hahn); torsten.schaub@uni-potsdam.de (T. Schaub)



© 2026 Copyright for this paper by its authors. Use permitted under Creative Commons License Attribution 4.0 International (CC BY 4.0).

CEUR Workshop Proceedings ceur-ws.org ISSN 1613-0073
published 2026-11-03

of logic programs [18] to this linear dynamic logic. More recently, [16] adopted the syntax of standard DL but applied it over a single sort of propositions to develop a semantics based on finite traces. A non-monotonic counterpart to this approach was subsequently introduced in [10, 11] by combining the approach with the logical characterization of ASP based on *Equilibrium Logic* [19]. By employing a single-sorted signature, both logics eliminate the traditional distinction between state and action variables. Although such a uniform treatment simplifies both the logical elaboration and computational evaluation, it is less intuitive from a knowledge representation perspective. This insight motivates us to advocate for a framework for reasoning about action and change that offers a rich modeling language, explicitly distinguishing between action and state variables, while remaining reducible to a uniform format for computational purposes.

More precisely, the paper is organized as follows: Section 2 introduces our formal framework, the *Dynamic Logic of Here-and-There with Actions and Fluents* (DHT^{af}). Within this section, Section 2.1 defines a variant of the two-sorted language from [8, 9] that integrates two logical systems for representing action and state formulas. Propositional logic describes combinations of actions that induce state transitions. These action formulas are then embedded within DL-based path expressions to formulate global constraints over entire trajectories, while state formulas are expressed using standard DL syntax. To accommodate a linear, non-monotonic semantics, we adapt the language from [8, 9] by omitting its functional restriction operator (since enforcing determinism is unnecessary in our linear-time approach) and by distinguishing between strong and default negation. Next, Section 2.2 accommodates the two-sorted signature by extending HT-traces [5] with transition labels comprising sets of primitive actions. Each set represents a classical model of the action formulas inducing the respective transition. The semantics of state formulas is defined as a DL-based extension of the Logic of Here-and-There [20] and its equilibrium models [21]. Finally, Section 2.3 establishes concepts of equivalence that guarantee the valid substitution of equivalent subformulas. Section 3 illustrates the modeling capacities of our framework for reasoning about action and change. We present the established modeling methodology of DL-based approaches alongside selected examples from the literature, and demonstrate how the prominent action language $\mathcal{BC}$ can be translated into our approach. Section 4 paves the way for implementation. We detail how formulas in our expressive language can be translated into the one-sorted dynamic logic underlying the temporal ASP solver *telingo* [22]. We detail the relationships to closely related approaches in Section 5. We conclude our paper in Section 6.

2. Dynamic Logic of Here-and-There with Actions and Fluents

2.1. Syntax

Following [8, 9], we consider a two-sorted alphabet consisting of disjoint non-empty sets $\mathcal{A}$ and $\mathcal{F}$ of action and state variables, respectively.

The variables in $\mathcal{A}$ represent *primitive actions*. In addition, we use verum ($\dot{\top}$) to allow for action-independent transitions. Action formulas describe *atomic actions* that cannot be broken into sequences of fewer actions; they amount to a set of primitive actions that are to be executed in parallel. Action formulas are expressed in terms of Boolean combinations of primitive actions. More precisely, the syntax of an *action formula* α over $\mathcal{A}$ is defined as follows.

$$\alpha ::= a \in \mathcal{A} \mid \dot{\bot} \mid \alpha \mathbin{\dot{\wedge}} \alpha \mid \dot{\neg}\alpha.$$

The Boolean operators are only decorated with dots to distinguish their use in action formulas from that in state formulas below, although logically analogous, they are treated semantically differently (see below). As we show below, falsum, disjunction and implication can be defined in the usual way, that is, $\dot{\top} \stackrel{\text{def}}{=} \dot{\neg}\dot{\bot}$, $\alpha \mathbin{\dot{\vee}} \beta \stackrel{\text{def}}{=} \dot{\neg}(\dot{\neg}\alpha \mathbin{\dot{\wedge}} \dot{\neg}\beta)$ and $\alpha \mathbin{\dot{\rightarrow}} \beta \stackrel{\text{def}}{=} \dot{\neg}(\alpha \mathbin{\dot{\wedge}} \dot{\neg}\beta)$.

The variables in $\mathcal{F}$ represent *primitive state properties*, also referred to as *fluents*. State formulas describe properties of states in terms of propositions and *complex actions* represented by path expressions. The syntax of a *state formula* φ and a *path expressions* ρ over $(\mathcal{A}, \mathcal{F})$ is defined as follows:

$$\varphi ::= f \in \mathcal{F} \mid \bot \mid {\sim}\varphi \mid \langle\rho\rangle\,\varphi \mid [\rho]\,\varphi \tag{1}$$

$$\rho ::= \alpha \mid \rho + \rho \mid \rho;\rho \mid \rho^* \mid \varphi? \mid \rho^- \tag{2}$$

where α is an action formula over $\mathcal{A}$. This syntax draws upon the one of Dynamic Logic [7]: $\langle\rho\rangle$ and $[\rho]$ are modal operators parametrized by a path expression ρ. More precisely, $\langle\rho\rangle\,\varphi$ expresses that after some execution of (possibly nondeterministic) complex action ρ property φ is satisfied and $[\rho]\,\varphi$ expresses that φ holds after each execution of ρ. Moreover, we use two types of negation: strong negation $\sim$ and a weaker form known as default negation, $\neg$. We rely on their interplay for expressing inertia in Section 3. Given our set $\mathcal{F}$ of fluents, we define the set $\widetilde{\mathcal{F}} = \{f, \sim f \mid f \in \mathcal{F}\}$. Finally, path expressions allow complex actions to compose atomic actions by nondeterministic choice ($\rho_1 + \rho_2$), sequential composition ($\rho_1;\rho_2$), reflexive transitive closure (ρ^*), inverse (ρ^-), and add a test ($\varphi?$), respectively. Path expressions allow for defining high-level control structures like if-then-else and while. For state formula φ and action formulas α, β, the conditional construct '*if* φ *then* α *else* β' is defined as $(\varphi?;\alpha) + (\sim\varphi?;\beta)$ and the loop construct '*while* φ *do* α' as $(\varphi?;\alpha)^*;\ \sim\varphi?$. As usual, a *theory* is a set of state formulas.

As we show below, this language allows us to capture several derived operators, like the Boolean and temporal ones:

$$\begin{aligned}
\varphi \wedge \psi &\stackrel{\text{def}}{=} \langle\varphi?\rangle\,\psi & \varphi \vee \psi &\stackrel{\text{def}}{=} \langle\varphi? + \psi?\rangle\,\top \\
\varphi \to \psi &\stackrel{\text{def}}{=} [\varphi?]\,\psi & \neg\varphi &\stackrel{\text{def}}{=} \varphi \to \bot \\
\varphi \leftrightarrow \psi &\stackrel{\text{def}}{=} [\varphi?]\,\psi \wedge [\psi?]\,\varphi & \varphi \Leftrightarrow \psi &\stackrel{\text{def}}{=} (\varphi \leftrightarrow \psi) \wedge (\sim\varphi \leftrightarrow \sim\psi) \\
\mathbb{F} &\stackrel{\text{def}}{=} [\dot{\top}]\bot & \mathbf{I} &\stackrel{\text{def}}{=} [\dot{\top}^-]\bot \\
\circ\varphi &\stackrel{\text{def}}{=} \langle\dot{\top}\rangle\,\varphi & \bullet\varphi &\stackrel{\text{def}}{=} \langle\dot{\top}^-\rangle\,\varphi \\
\widehat{\circ}\varphi &\stackrel{\text{def}}{=} [\dot{\top}]\,\varphi & \widehat{\bullet}\varphi &\stackrel{\text{def}}{=} [\dot{\top}^-]\,\varphi \\
\Diamond\varphi &\stackrel{\text{def}}{=} \langle\dot{\top}^*\rangle\,\varphi & \blacklozenge\varphi &\stackrel{\text{def}}{=} \langle\dot{\top}^{-*}\rangle\,\varphi \\
\Box\varphi &\stackrel{\text{def}}{=} [\dot{\top}^*]\,\varphi & \blacksquare\varphi &\stackrel{\text{def}}{=} [\dot{\top}^{-*}]\,\varphi \\
\varphi\,\mathbb{U}\,\psi &\stackrel{\text{def}}{=} \langle(\varphi?;\dot{\top})^*\rangle\,\psi & \varphi\,\mathbf{S}\,\psi &\stackrel{\text{def}}{=} \langle(\varphi?;\dot{\top}^-)^*\rangle\,\psi \\
\varphi\,\mathbb{R}\,\psi &\stackrel{\text{def}}{=} (\psi\,\mathbb{U}\,(\varphi \wedge \psi)) \vee \Box\psi & \varphi\,\mathbf{T}\,\psi &\stackrel{\text{def}}{=} (\psi\,\mathbf{S}\,(\varphi \wedge \psi)) \vee \blacksquare\psi
\end{aligned}$$

All connectives are defined in terms of the dynamic operators $\langle\cdot\rangle$ and $[\cdot]$. This involves the Booleans $\wedge$, $\vee$, and $\to$. Among them, the definition of $\to$ is most noteworthy since it hints at the implicative nature of $[\cdot]$. Negation $\neg$ is then expressed via implication, as usual in the Logic of Here-and-There (HT; [20]). Then, $\langle\cdot\rangle$ and $[\cdot]$ also allow for defining the future temporal operators $\mathbb{F}$, $\circ$, $\widehat{\circ}$, $\Diamond$, $\Box$, $\mathbb{U}$, $\mathbb{R}$, standing for *final, next, weak next, eventually, always, until,* and *release*, and their past-oriented counterparts: $\mathbf{I}$, $\bullet$, $\widehat{\bullet}$, $\blacklozenge$, $\blacksquare$, $\mathbf{S}$, $\mathbf{T}$. The weak one-step operators, $\widehat{\circ}$ and $\widehat{\bullet}$, are of particular interest when dealing with finite traces, since their behavior differs from their genuine counterparts only at the ends of a trace. In fact, $\widehat{\circ}\varphi$ can also be expressed as $\circ\varphi \vee \mathbb{F}$ (and $\widehat{\bullet}$ as $\bullet\varphi \vee \mathbf{I}$). Finally, note that the converse operator ρ^- is essential for expressing all temporal past operators, whose addition in temporal logic is exponentially more succinct than using only future operators [23]. A formula is *propositional*, if all its connectives are Boolean, and *temporal*, if it includes only Boolean and temporal ones.

2.2. Semantics

We define the semantics of state formulas in terms of *traces* over $\widetilde{\mathcal{F}}$, which are sequences $\mathbf{T} = (T_i)_{i\in[0..\lambda)}$ of (consistent) interpretations $T_i \subseteq \widetilde{\mathcal{F}}$ satisfying $\{f, \sim f\} \not\subseteq T_i$ for all $i \in [0..\lambda)$ and $f \in \mathcal{F}$; λ is the length of $\mathbf{T}$. Here, we consider only traces of finite length. A trace $\mathbf{T}$ is X-complete for $X \subseteq \mathcal{F}$, if either $f \in T_i$ or $\sim f \in T_i$ for all $i \in [0..\lambda)$ and $f \in X$. We define the ordering $\mathbf{H} \leq \mathbf{T}$ between traces of the same length λ as $H_i \subseteq T_i$ 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 $\widetilde{\mathcal{F}}$ of length λ 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 all $i \in [0..\lambda)$. For convenience, we sometimes represent it as the pair $\langle\mathbf{H}, \mathbf{T}\rangle$ of traces $\mathbf{H} = (H_i)_{i\in[0..\lambda)}$ and $\mathbf{T} = (T_i)_{i\in[0..\lambda)}$. Actions are associated with the transitions in a trace via a function $\aleph$. A *labeled HT-trace* of length λ over $(\mathcal{A}, \widetilde{\mathcal{F}})$ is a pair $(\langle\mathbf{H}, \mathbf{T}\rangle, \aleph)$ consisting of

1. an HT-trace $\langle \mathbf{H}, \mathbf{T} \rangle = (\langle H_i, T_i \rangle)_{i \in [0..\lambda)}$ over $\widetilde{\mathcal{F}}$ and
2. a function $\aleph : \mathcal{A} \to 2^{I_\lambda}$ where $I_\lambda = \{(i, i+1) \mid i, i+1 \in [0..\lambda)\}$

As an example, consider a labeled trace with

$$\big(\langle \emptyset, \{p\}\rangle, \langle \{p\}, \{p,q\}\rangle, \langle \{q\}, \{p,q\}\rangle\big) \qquad \aleph(a) = \{(0,1),(1,2)\} \quad \aleph(b) = \{(0,1)\}$$

A labeled HT-trace $\mathbf{M} = (\langle \mathbf{H}, \mathbf{T} \rangle, \aleph)$ of length λ over $(\mathcal{A}, \widetilde{\mathcal{F}})$ satisfies (resp. falsifies) a state formula φ over $(\mathcal{A}, \mathcal{F})$ at $i \in [0..\lambda)$, in symbols $\mathbf{M}, i \models \varphi$ (resp. $\mathbf{M}, i \models\!\!\!| \, \varphi$), if the following conditions hold:

1. $\mathbf{M}, i \models f$ if $f \in H_i$ for $f \in \mathcal{F}$
2. $\mathbf{M}, i \not\models \bot$
3. $\mathbf{M}, i \models \sim\!\psi$ if $\mathbf{M}, i \models\!\!\!| \, \psi$
4. $\mathbf{M}, i \models \langle \rho \rangle \, \varphi$ if $\mathbf{M}, j \models \varphi$ for some $(i,j) \in \|\rho\|^{\mathbf{M}}_{\models}$
5. $\mathbf{M}, i \models [\rho] \, \varphi$ if $(\langle \mathbf{X}, \mathbf{T} \rangle, \aleph), j \models \varphi$ for all $(i,j) \in \|\rho\|^{(\langle \mathbf{X}, \mathbf{T} \rangle, \aleph)}_{\models}$ and $\mathbf{X} \in \{\mathbf{H}, \mathbf{T}\}$
6. $\mathbf{M}, i \models\!\!\!| \, f$ if $\sim\!f \in H_i$ for $f \in \mathcal{F}$
7. $\mathbf{M}, i \models\!\!\!| \, \bot$
8. $\mathbf{M}, i \models\!\!\!| \, \sim\!\psi$ if $\mathbf{M}, i \models \psi$
9. $\mathbf{M}, i \models\!\!\!| \, \langle \rho \rangle \, \varphi$ if $\mathbf{M}, j \models\!\!\!| \, \varphi$ for all $(i,j) \in \|\rho\|^{\mathbf{M}}_{\not\models}$
10. $\mathbf{M}, i \models\!\!\!| \, [\rho] \, \varphi$ if $\mathbf{M}, j \models\!\!\!| \, \varphi$ for some $(i,j) \in \|\rho\|^{\mathbf{M}}_{\models}$

where the relations $\|\rho\|^{\mathbf{M}}_{\models} \subseteq [0..\lambda) \times [0..\lambda)$ and $\|\rho\|^{\mathbf{M}}_{\not\models} \subseteq [0..\lambda) \times [0..\lambda)$ are defined for action variable $a \in \mathcal{A}$, action formulas α, β, state formula φ, and path expressions ρ, ϱ, as

$$\begin{aligned}
\|\dot{\bot}\|^{\mathbf{M}}_{\models} &\stackrel{\text{def}}{=} \emptyset & \|\dot{\bot}\|^{\mathbf{M}}_{\not\models} &\stackrel{\text{def}}{=} \emptyset \\
\|a\|^{\mathbf{M}}_{\models} &\stackrel{\text{def}}{=} \aleph(a) & \|a\|^{\mathbf{M}}_{\not\models} &\stackrel{\text{def}}{=} \aleph(a) \\
\|\alpha \mathbin{\dot{\wedge}} \beta\|^{\mathbf{M}}_{\models} &\stackrel{\text{def}}{=} \|\alpha\|^{\mathbf{M}}_{\models} \cap \|\beta\|^{\mathbf{M}}_{\models} & \|\alpha \mathbin{\dot{\wedge}} \beta\|^{\mathbf{M}}_{\not\models} &\stackrel{\text{def}}{=} \|\alpha\|^{\mathbf{M}}_{\not\models} \cap \|\beta\|^{\mathbf{M}}_{\not\models} \\
\|\dot{\neg}\alpha\|^{\mathbf{M}}_{\models} &\stackrel{\text{def}}{=} I_\lambda \setminus \|\alpha\|^{\mathbf{M}}_{\models} & \|\dot{\neg}\alpha\|^{\mathbf{M}}_{\not\models} &\stackrel{\text{def}}{=} I_\lambda \setminus \|\alpha\|^{\mathbf{M}}_{\not\models} \\
\|\varphi?\|^{\mathbf{M}}_{\models} &\stackrel{\text{def}}{=} \{(i,i) \mid \mathbf{M}, i \models \varphi\} & \|\varphi?\|^{\mathbf{M}}_{\not\models} &\stackrel{\text{def}}{=} \{(i,i) \mid \mathbf{M}, i \not\models\!\!\!| \, \varphi\} \\
\|\rho^-\|^{\mathbf{M}}_{\models} &\stackrel{\text{def}}{=} \{(k,i) \mid (i,k) \in \|\rho\|^{\mathbf{M}}_{\models}\} & \|\rho^-\|^{\mathbf{M}}_{\not\models} &\stackrel{\text{def}}{=} \{(k,i) \mid (i,k) \in \|\rho\|^{\mathbf{M}}_{\not\models}\} \\
\|\rho + \varrho\|^{\mathbf{M}}_{\models} &\stackrel{\text{def}}{=} \|\rho\|^{\mathbf{M}}_{\models} \cup \|\varrho\|^{\mathbf{M}}_{\models} & \|\rho + \varrho\|^{\mathbf{M}}_{\not\models} &\stackrel{\text{def}}{=} \|\rho\|^{\mathbf{M}}_{\not\models} \cup \|\varrho\|^{\mathbf{M}}_{\not\models} \\
\|\rho \mathbin{;} \varrho\|^{\mathbf{M}}_{\models} &\stackrel{\text{def}}{=} \|\rho\|^{\mathbf{M}}_{\models} \circ \|\varrho\|^{\mathbf{M}}_{\models} & \|\rho \mathbin{;} \varrho\|^{\mathbf{M}}_{\not\models} &\stackrel{\text{def}}{=} \|\rho\|^{\mathbf{M}}_{\not\models} \circ \|\varrho\|^{\mathbf{M}}_{\not\models} \\
\|\rho^*\|^{\mathbf{M}}_{\models} &\stackrel{\text{def}}{=} \Big(\|\rho\|^{\mathbf{M}}_{\models}\Big)^* & \|\rho^*\|^{\mathbf{M}}_{\not\models} &\stackrel{\text{def}}{=} \Big(\|\rho\|^{\mathbf{M}}_{\not\models}\Big)^*
\end{aligned}$$

Note that the difference between $\|\rho\|^{\mathbf{M}}_{\models}$ and $\|\rho\|^{\mathbf{M}}_{\not\models}$ boils down to their treatment of test formulas '$\varphi?$'.

A labeled HT-trace $\mathbf{M}$ is a *model* of a state formula φ if $\mathbf{M}, 0 \models \varphi$, and it is a *model* of a theory Γ if $\mathbf{M}, 0 \models \varphi$ for all $\varphi \in \Gamma$. A formula φ is a tautology, written $\models \varphi$, iff $\mathbf{M}, k \models \varphi$ for any labeled HT-trace $\mathbf{M}$ and any $k \in [0..\lambda)$. We call the logic induced by the set of all tautologies *Dynamic logic of Here-and-There with Actions and Fluents* (DHT^{af} for short).

Following [24], two formulas φ and ψ are *weakly equivalent* (written $\varphi \equiv \psi$) if $\mathbf{M}, k \models \varphi$ iff $\mathbf{M}, k \models \psi$, and *strictly equivalent* (written $\varphi \cong \psi$) if they additionally satisfy $\mathbf{M}, k \models\!\!\!| \, \varphi$ iff $\mathbf{M}, k \models\!\!\!| \, \psi$ for any labeled HT-trace $\mathbf{M}$ and any $k \in [0..\lambda)$. Both notions of equivalence are stronger than the mere coincidence of models, since models are evaluated only at the initial state. Analogously, two path expressions ρ and ρ' are *weakly equivalent* (written $\rho \equiv \rho'$) if $\|\rho\|^{\mathbf{M}}_{\models} = \|\rho'\|^{\mathbf{M}}_{\models}$, and *strictly equivalent* (written $\rho \cong \rho'$) if they additionally satisfy $\|\rho\|^{\mathbf{M}}_{\not\models} = \|\rho'\|^{\mathbf{M}}_{\not\models}$ for any labeled HT-trace $\mathbf{M}$.

A labeled HT-trace of form $(\langle \mathbf{T}, \mathbf{T} \rangle, \aleph)$ is an *equilibrium model* of a theory Γ if it is a model of Γ and there is no other model $(\langle \mathbf{H}, \mathbf{T} \rangle, \aleph)$ of Γ with $\mathbf{H} < \mathbf{T}$. Then, we also say that $\mathbf{T}$ is a *stable model* of Γ.

2.3. Elaboration

With the structure of labeled HT-traces established, we can now characterize the specific set of primitive actions that induce each state transition. For a labeled HT-trace $(\langle \mathbf{H}, \mathbf{T} \rangle, \aleph)$ of length λ over $(\mathcal{A}, \widetilde{\mathcal{F}})$, we define $A_{(i,i+1)} = \{a \in \mathcal{A} \mid (i, i+1) \in \aleph(a)\}$ for $i, i+1 \in [0..\lambda)$. The set $A_{(i,i+1)}$ contains all primitive actions associated with the transition from state i to $i+1$.

Proposition 1. *Let* $\mathbf{M}$ *be a labeled HT-trace of length* λ *over* $(\mathcal{A}, \widetilde{\mathcal{F}})$ *and* α *an action formula over* $\mathcal{A}$. *For any* $i, i{+}1 \in [0..\lambda)$ *and* $\otimes \in \{\models, \not\models\}$, *we have* $(i, i{+}1) \in \|\alpha\|^{\mathbf{M}}_{\otimes}$ *iff* $A_{(i,i+1)}$ *is a classical model of* α.

The set of primitive actions inducing a state transition forms a classical model entailing all action formulas associated with this transition. Accordingly, all (dotted) action formulas are treated classically.

For arbitrary path expressions, we establish the following relationship.

Proposition 2. *Let* ρ *be a path expression and* $(\langle \mathbf{H}, \mathbf{T} \rangle, \aleph)$ *be a labeled HT trace over* $(\mathcal{A}, \widetilde{\mathcal{F}})$. *Then, we have that* $\|\rho\|^{(\langle \mathbf{H},\mathbf{T}\rangle,\aleph)}_{\models} \subseteq \|\rho\|^{(\langle \mathbf{H},\mathbf{T}\rangle,\aleph)}_{\not\models}$.

This subset relation becomes an equality whenever ρ is test-free, allowing $\langle \mathbf{H}, \mathbf{T} \rangle$ and $\langle \mathbf{T}, \mathbf{T} \rangle$ to be treated interchangeably.

Let us now turn to some properties of the satisfaction and falsification relations.

Proposition 3. *Let* φ *be a state formula,* ρ *a path expression, and* $(\langle \mathbf{H}, \mathbf{T} \rangle, \aleph)$ *be a labeled HT trace of length* λ. *Then, we have for all* $i \in [0..\lambda)$ *that*

1. $(\langle \mathbf{H}, \mathbf{T} \rangle, \aleph), i \models \varphi$ *implies* $(\langle \mathbf{T}, \mathbf{T} \rangle, \aleph), i \models \varphi$
2. $(\langle \mathbf{H}, \mathbf{T} \rangle, \aleph), i \not\models \varphi$ *implies* $(\langle \mathbf{T}, \mathbf{T} \rangle, \aleph), i \not\models \varphi$
3. $\|\rho\|^{(\langle \mathbf{H},\mathbf{T}\rangle,\aleph)}_{\models} \subseteq \|\rho\|^{(\langle \mathbf{T},\mathbf{T}\rangle,\aleph)}_{\models}$
4. $\|\rho\|^{(\langle \mathbf{H},\mathbf{T}\rangle,\aleph)}_{\not\models} \supseteq \|\rho\|^{(\langle \mathbf{T},\mathbf{T}\rangle,\aleph)}_{\not\models}$

Corollary 1. *Let* φ *be a state formula,* ρ *a path expression, and* $(\langle \mathbf{H}, \mathbf{T} \rangle, \aleph)$ *be a labeled HT trace of length* λ. *Then, we have for all* $i \in [0..\lambda)$ *that*

1. $(\langle \mathbf{H}, \mathbf{T} \rangle, \aleph), i \models \neg\varphi$ *iff* $(\langle \mathbf{T}, \mathbf{T} \rangle, \aleph), i \not\models \varphi$
2. $(\langle \mathbf{H}, \mathbf{T} \rangle, \aleph), i \not\models \neg\varphi$ *iff* $(\langle \mathbf{H}, \mathbf{T} \rangle, \aleph), i \models \varphi$

The next result demonstrates that our core language embeds all Boolean and linear-time temporal operators with their standard semantics.

Theorem 1. *Let* $\mathbf{M} = (\langle \mathbf{H}, \mathbf{T} \rangle, \aleph)$ *be a labeled HT-trace of length* λ *over* $(\mathcal{A}, \widetilde{\mathcal{F}})$. *Given the respective definitions of derived operators, we get the following satisfaction conditions for all* $k \in [0..\lambda)$*:*

1. $\mathbf{M}, k \models \varphi \wedge \psi$ *iff* $\mathbf{M}, k \models \varphi$ *and* $\mathbf{M}, k \models \psi$
2. $\mathbf{M}, k \models \varphi \vee \psi$ *iff* $\mathbf{M}, k \models \varphi$ *or* $\mathbf{M}, k \models \psi$
3. $\mathbf{M}, k \models \varphi \rightarrow \psi$ *iff* $\mathbf{M}', k \not\models \varphi$ *or* $\mathbf{M}', k \models \psi$ *for all* $\mathbf{M}' \in \{\mathbf{M}, (\langle \mathbf{T}, \mathbf{T} \rangle, \aleph)\}$
4. $\mathbf{M}, k \models \varphi \leftrightarrow \psi$ *iff* $\mathbf{M}', k \models \varphi$ *iff* $\mathbf{M}', k \models \psi$ *for all* $\mathbf{M}' \in \{\mathbf{M}, (\langle \mathbf{T}, \mathbf{T} \rangle, \aleph)\}$
5. $\mathbf{M}, k \models \varphi \Leftrightarrow \psi$ *iff* $\mathbf{M}', k \models \varphi$ *iff* $\mathbf{M}', k \models \psi$, *and* $\mathbf{M}', k \not\models \varphi$ *iff* $\mathbf{M}', k \not\models \psi$ *for all* $\mathbf{M}' \in \{\mathbf{M}, (\langle \mathbf{T}, \mathbf{T} \rangle, \aleph)\}$
6. $\mathbf{M}, k \models \neg\varphi$ *iff* $(\langle \mathbf{T}, \mathbf{T} \rangle, \aleph), k \not\models \varphi$
7. $\mathbf{M}, k \models \mathbb{F}$ *iff* $k + 1 = \lambda$
8. $\mathbf{M}, k \models \circ\varphi$ *iff* $k + 1 < \lambda$ *and* $\mathbf{M}, k{+}1 \models \varphi$
9. $\mathbf{M}, k \models \widehat{\circ}\varphi$ *iff* $k + 1 = \lambda$ *or* $\mathbf{M}, k{+}1 \models \varphi$
10. $\mathbf{M}, k \models \Diamond\varphi$ *iff* $\mathbf{M}, i \models \varphi$ *for some* $i \in [k..\lambda)$
11. $\mathbf{M}, k \models \Box\varphi$ *iff* $\mathbf{M}, i \models \varphi$ *for all* $i \in [k..\lambda)$
12. $\mathbf{M}, k \models \varphi \mathbb{U} \psi$ *iff for some* $j \in [k..\lambda)$, *we have* $\mathbf{M}, j \models \psi$ *and* $\mathbf{M}, i \models \varphi$ *for all* $i \in [k..j)$
13. $\mathbf{M}, k \models \varphi \mathbb{R} \psi$ *iff for all* $j \in [k..\lambda)$, *we have* $\mathbf{M}, j \models \psi$ *or* $\mathbf{M}, i \models \varphi$ *for some* $i \in [k..j)$
14. $\mathbf{M}, k \models \mathsf{I}$ *iff* $k = 0$
15. $\mathbf{M}, k \models \bullet\varphi$ *iff* $k > 0$ *and* $\mathbf{M}, k{-}1 \models \varphi$
16. $\mathbf{M}, k \models \widehat{\bullet}\varphi$ *iff* $k = 0$ *or* $\mathbf{M}, k{-}1 \models \varphi$
17. $\mathbf{M}, k \models \blacksquare\varphi$ *iff* $\mathbf{M}, i \models \varphi$ *for all* $i \in [0..k]$
18. $\mathbf{M}, k \models \blacklozenge\varphi$ *iff* $\mathbf{M}, i \models \varphi$ *for some* $i \in [0..k]$

19. $\mathbf{M}, k \models \varphi \,\mathbf{S}\, \psi$ *iff for some* $j \in [0..k]$*, we have* $\mathbf{M}, j \models \psi$ *and* $\mathbf{M}, i \models \varphi$ *for all* $i \in [j+1..k]$
20. $\mathbf{M}, k \models \varphi \,\mathbf{T}\, \psi$ *iff for all* $j \in [0..k]$*, we have* $\mathbf{M}, j \models \psi$ *or* $\mathbf{M}, i \models \varphi$ *for some* $i \in [j+1..k]$

as well as the relation:

21. $\|\alpha \dot{\vee} \beta\|^{\mathbf{M}}_{\models} = \|\dot{\neg}(\dot{\neg}\alpha \dot{\wedge} \dot{\neg}\beta)\|^{\mathbf{M}}_{\models}$
22. $\|\alpha \dot{\rightarrow} \beta\|^{\mathbf{M}}_{\models} = \|\dot{\neg}(\alpha \dot{\wedge} \dot{\neg}\beta)\|^{\mathbf{M}}_{\models}$

Satisfaction conditions 4 and 5 show that $\varphi \equiv \psi$ and $\varphi \cong \psi$ amount to requiring that $\varphi \leftrightarrow \psi$ and $\varphi \Leftrightarrow \psi$ are tautologies, respectively. Notably, the condition $\mathbf{M}, k \dashv \varphi$ iff $\mathbf{M}, k \dashv \psi$ in 5 can be restated as $\mathbf{M}, k \models \sim\varphi$ iff $\mathbf{M}, k \models \sim\psi$. Furthermore, conditions 7 through 20 coincide with those for the temporal operators in Linear-time Logic of Here-and-There (THT; [25]).

The following collection of propositions show how satisfaction is mirrored by falsification.

Theorem 2. *Let* $\mathbf{M} = (\langle \mathbf{H}, \mathbf{T} \rangle, \aleph)$ *be a labeled HT-trace of length* λ *over* $(\mathcal{A}, \widetilde{\mathcal{F}})$*. Given the respective definitions of derived operators, we get the following falsification conditions for all* $k \in [0..\lambda)$*:*

1. $\mathbf{M}, k \dashv \varphi \wedge \psi$ *iff* $\mathbf{M}, k \dashv \varphi$ *or* $\mathbf{M}, k \dashv \psi$
2. $\mathbf{M}, k \dashv \varphi \vee \psi$ *iff* $\mathbf{M}, k \dashv \varphi$ *and* $\mathbf{M}, k \dashv \psi$
3. $\mathbf{M}, k \dashv \varphi \rightarrow \psi$ *iff* $\mathbf{M}, k \models \varphi$ *and* $\mathbf{M}, k \dashv \psi$
4. $\mathbf{M}, k \dashv \varphi \leftrightarrow \psi$ *iff either* $\mathbf{M}, k \models \varphi$ *and* $\mathbf{M}, k \dashv \psi$ *or* $\mathbf{M}, k \models \psi$ *and* $\mathbf{M}, k \dashv \varphi$
5. $\mathbf{M}, k \dashv \varphi \Leftrightarrow \psi$ *iff either* $\mathbf{M}, k \models \varphi$ *and* $\mathbf{M}, k \dashv \psi$ *or* $\mathbf{M}, k \models \psi$ *and* $\mathbf{M}, k \dashv \varphi$
6. $\mathbf{M}, k \dashv \neg\varphi$ *iff* $\mathbf{M}, k \models \varphi$
7. $\mathbf{M}, k \dashv \mathbb{F}$ *iff* $k + 1 < \lambda$
8. $\mathbf{M}, k \dashv \circ\varphi$ *iff* $k + 1 = \lambda$ *or* $\mathbf{M}, k{+}1 \dashv \varphi$
9. $\mathbf{M}, k \dashv \widehat{\circ}\varphi$ *iff* $k + 1 < \lambda$ *and* $\mathbf{M}, k{+}1 \dashv \varphi$
10. $\mathbf{M}, k \dashv \Diamond\varphi$ *iff* $\mathbf{M}, i \dashv \varphi$ *for all* $i \in [k..\lambda)$
11. $\mathbf{M}, k \dashv \Box\varphi$ *iff* $\mathbf{M}, i \dashv \varphi$ *for some* $i \in [k..\lambda)$
12. $\mathbf{M}, k \dashv \varphi \,\mathbb{U}\, \psi$ *iff for all* $j \in [k..\lambda)$*, we have* $\mathbf{M}, j \dashv \psi$ *or* $\mathbf{M}, i \dashv \varphi$ *for some* $i \in [k..j)$
13. $\mathbf{M}, k \dashv \varphi \,\mathbb{R}\, \psi$ *iff for some* $j \in [k..\lambda)$*, we have* $\mathbf{M}, j \dashv \psi$ *and* $\mathbf{M}, i \dashv \varphi$ *for all* $i \in [k..j)$
14. $\mathbf{M}, k \dashv \mathbf{I}$ *iff* $k > 0$
15. $\mathbf{M}, k \dashv \bullet\varphi$ *iff* $k = 0$ *or* $\mathbf{M}, k{-}1 \dashv \varphi$
16. $\mathbf{M}, k \dashv \widehat{\bullet}\varphi$ *iff* $k > 0$ *and* $\mathbf{M}, k{-}1 \dashv \varphi$
17. $\mathbf{M}, k \dashv \blacksquare\varphi$ *iff* $\mathbf{M}, i \dashv \varphi$ *for some* $i \in [0..k]$
18. $\mathbf{M}, k \dashv \blacklozenge\varphi$ *iff* $\mathbf{M}, i \dashv \varphi$ *for all* $i \in [0..k]$
19. $\mathbf{M}, k \dashv \varphi \,\mathbf{S}\, \psi$ *iff for all* $j \in [0..k]$*, we have* $\mathbf{M}, j \dashv \psi$ *or* $\mathbf{M}, i \dashv \varphi$ *for some* $i \in (j+1..k]$
20. $\mathbf{M}, k \dashv \varphi \,\mathbf{T}\, \psi$ *iff for some* $j \in [0..k]$*, we have* $\mathbf{M}, j \dashv \psi$ *and* $\mathbf{M}, i \dashv \varphi$ *for some* $i \in (j..k]$

as well as the relation:

21. $\|\alpha \dot{\vee} \beta\|^{\mathbf{M}}_{\not\models} = \|\dot{\neg}(\dot{\neg}\alpha \dot{\wedge} \neg\beta)\|^{\mathbf{M}}_{\not\models}$
22. $\|\alpha \dot{\rightarrow} \beta\|^{\mathbf{M}}_{\not\models} = \|\dot{\neg}(\alpha \dot{\wedge} \neg\beta)\|^{\mathbf{M}}_{\not\models}$

Interestingly, connectives $\leftrightarrow$ and $\Leftrightarrow$ are indistinguishable in terms of falsification.

Despite the strict separation of actions and fluents, DL can induce transitions from state formulas ϕ via the path expression '$\phi?; \dot{\top}$'. This mechanism extends to our approach, as formalized below.

Proposition 4. *Let* $\mathbf{M}$ *be a labeled HT-trace of length* λ *and* ϕ *be a state formula. Then, we have*

1. $\|\phi?; \dot{\top}\|^{\mathbf{M}}_{\models} = \{(i, i{+}1) \mid \mathbf{M}, i \models \phi, i \in [0..\lambda{-}1)\}$
2. $\|\phi?; \dot{\top}\|^{\mathbf{M}}_{\not\models} = \{(i, i{+}1) \mid \mathbf{M}, i \not\models \phi, i \in [0..\lambda{-}1)\}$

We leverage this property in Section 4 to translate two-sorted state formulas of DEL^{af} into one-sorted formulas of DEL.

Next, we show that incorporating the law of the excluded middle yields total models, reflecting non-constructive (temporal) logical systems.

Proposition 5. *Let* $(\langle \mathbf{H}, \mathbf{T} \rangle, \aleph)$ *be a labeled HT-trace over* $(\mathcal{A}, \widetilde{\mathcal{F}})$.
Then, $(\langle \mathbf{H}, \mathbf{T} \rangle, \aleph)$ *is a model of* $\{ [\dot{\top}^*] (f \vee \neg f) \mid f \in \widetilde{\mathcal{F}} \}$ *iff* $\mathbf{H} = \mathbf{T}$.

The addition of these axioms collapses DEL^{af} into a temporal variant of *classical logic with strong negation* [26],[1] where the truth of fluents can remain undefined. To eliminate this indeterminate value and enforce that every fluent is strictly true or false, we require a stronger version of these axioms.

Proposition 6. *Let* $(\langle \mathbf{H}, \mathbf{T} \rangle, \aleph)$ *be a labeled HT-trace over* $(\mathcal{A}, \widetilde{\mathcal{F}})$.
Then, $(\langle \mathbf{H}, \mathbf{T} \rangle, \aleph)$ *is a model of* $\{ [\dot{\top}^*] (f \vee {\sim} f) \mid f \in \mathcal{F} \}$ *iff* $\mathbf{H} = \mathbf{T}$ *and* $(\langle \mathbf{H}, \mathbf{T} \rangle, \aleph)$ *is* $\mathcal{F}$*-complete.*

Finally, we show how the equivalence concepts established above guarantee the valid substitution of equivalent formulas and path expressions.

Proposition 7. *Let* φ *and* φ' *be two state formulas. If* $\varphi \cong \varphi'$*, then* $\varphi? \cong \varphi'?$.

Proposition 8. *Let* φ *be a state formula and let* ρ *and* ρ' *be two path expressions.*

1. *If* $\rho \cong \rho'$*, then* $\langle \rho \rangle \varphi \cong \langle \rho' \rangle \varphi$.
2. *If* $\rho \cong \rho'$*, then* $[\rho] \varphi \cong [\rho'] \varphi$.

In what follows, we refer to both state formulas and path expressions collectively as *expressions*. Given an expression ε and two expressions ϵ and ϵ' of the same type, let $\varepsilon[\epsilon/\epsilon']$ denote the expression obtained by replacing all occurrences of ϵ in ε with ϵ'.

Theorem 3. *Let* ε *be an expression, and let* ϵ *and* ϵ' *be two expressions of the same type. If* $\epsilon \cong \epsilon'$*, then* $\varepsilon \cong \varepsilon[\epsilon/\epsilon']$.

As usual, two theories Γ and Γ' are *strongly equivalent* if the equilibrium models of $\Gamma \cup \Delta$ and $\Gamma' \cup \Delta$ coincide for any theory Δ. This ensures that Γ can be safely replaced by Γ' in any common context Δ without altering the set of equilibrium models. The following result shows that checking strong equivalence in DEL^{af} reduces to weak equivalence in the monotonic logic DHT^{af}.

Theorem 4. *Let* Γ *and* Γ' *be two theories. Then,* Γ *and* Γ' *are strongly equivalent iff* $\Gamma \equiv \Gamma'$.

Similarly, two expressions ϵ and ϵ' of the same type are *strongly equivalent on substitutions* if the equilibrium models of $\{\varphi[\epsilon/\epsilon']\} \cup \Delta$ and $\{\varphi\} \cup \Delta$ coincide for any theory Δ and state formula φ. Unlike strong equivalence, checking substitution equivalence in DEL^{af} reduces to strict equivalence in DHT^{af}.

Theorem 5. *Let* ϵ *and* ϵ' *be two expressions of the same type. Then,* ϵ *and* ϵ' *are strongly equivalent on substitutions iff* $\epsilon \cong \epsilon'$.

3. Reasoning about Action and Change

Following [8, 9], we view action formulas as defining state transitions. Each transition is determined by a set of literals that specify the simultaneous execution, or non-execution, of certain actions. Any actions not explicitly specified are considered exogenous and may freely vary. In this framework, the conjunction $\alpha \mathbin{\dot{\wedge}} \beta$ of atomic actions α and β enforces their simultaneous execution, whereas the negation $\dot{\neg}\alpha$ strictly prohibits the execution of α. Furthermore, the disjunction $\alpha \mathbin{\dot{\vee}} \beta$ represents the non-deterministic execution of α or β, effectively providing a non-deterministic choice between state transitions. Conversely, the formula $\dot{\neg}\alpha \mathbin{\dot{\vee}} \dot{\neg}\beta$ expresses that α and β cannot be executed simultaneously.

[1] Classical logic with strong negation extends standard classical logic by introducing an independent negation operator to capture explicit falsity, rather than the mere absence of truth. This allows reasoning systems to distinguish between being "not true" and "explicitly false," thereby enriching classical logic while preserving much of its original structure. In this setting, an atom f may take one of three truth values: true, false, or undefined.

Building on these atomic transitions, path expressions allow us to describe complex sequences of state transitions. For instance, the formula $[\rho]\,\varphi$ requires that the state formula φ holds in all states reachable by any action sequence expressed by the path expression ρ. Conversely, $[\rho]\,\bot$ expresses the inability to execute ρ. Additionally, combining path expressions with action negation, such as $[\dot{\neg}\alpha]\,\bot$, allows us to concisely express the inability to perform any atomic actions other than those in α. Complementary to the always operator, which dictates behavior across all possible paths, the dual eventually operator expresses existential properties over action sequences. Specifically, $\langle\rho\rangle\,\varphi$ guarantees that a valid execution of ρ leads to at least one state satisfying φ. Consequently, the formula $\langle\rho\rangle\,\top$ strictly mandates the actual execution of ρ.[2]

These modal literals serve as the foundational building blocks for modeling actions and change within Dynamic Logic. To systematize this, we adopt the taxonomy of action laws proposed by [13]. A *static law* is a state formula φ over $\mathcal{F}$ of the form $\Box\varphi$. An *effect law* for an action formula α takes the form $\Box(\varphi \rightarrow [\alpha]\,\psi)$, where φ and ψ are state formulas, Here, the consequent ψ is the effect obtained when the action α is executed in a state satisfying the precondition φ. An *inexecutability law* for an action formula α is of the form $\Box(\varphi \rightarrow [\alpha]\,\bot)$, denoting that α cannot be executed in states where state formula φ holds.[3] Finally, the non-monotonicity of our approach offers a succinct mechanism to capture the default nature of inertia, without requiring any explicit reference to action formulas. As put forward by [4], a *persistency law* for a state formula φ can be expressed as $\Box(\varphi \wedge \neg\circ{\sim}\varphi \rightarrow \widehat{\circ}\varphi)$. Intuitively, this dictates that the satisfaction of φ remains unchanged upon a state transition, unless there is a reason for it to change in the subsequent state.

3.1. Parallelism at work

In this section, we demonstrate the modeling advantages of our framework through representative examples, focusing on its capacity to specify concurrency and complex control rules.

The first example deals with lifting a bowl of soup [27, 28]: *Whenever Mary tries to lift the bowl with one hand, she spills the soup. When she uses both hands, she does not spill the soup. We know that the soup is not spilled initially.* Formalizing this scenario requires accounting for dependencies between actions. In particular, we utilize conjunction ($\dot{\wedge}$) to express joint execution and negation ($\dot{\neg}$) to represent non-execution of actions. We represent the domain using the actions ll and lr that denote lifting the bowl with the left and right hand, respectively, and the fluent $empty$ to indicate that the bowl is empty. The fact that the bowl is full is represented by the literal ${\sim}empty$. We model this example as follows:

$$
\begin{aligned}
{\sim}empty &&(3)\\
\Box({\sim}empty \rightarrow [ll\ \dot{\wedge}\ lr]\,{\sim}empty) &&(4)\\
\Box(\top \rightarrow [(ll\ \dot{\wedge}\ \dot{\neg}lr)\ \dot{\vee}\ (lr\ \dot{\wedge}\ \dot{\neg}ll)]\,empty) &&(5)\\
\Box(empty \wedge \neg\circ{\sim}empty \rightarrow \widehat{\circ}empty) &&(6)\\
\Box({\sim}empty \wedge \neg\circ empty \rightarrow \widehat{\circ}{\sim}empty). &&(7)
\end{aligned}
$$

Rule (3) establishes that the bowl is initially full. Effect laws (4) and (5) characterize the consequences of the actions ll and lr. Specifically, Rule (4) states that the bowl remains full if both actions are performed concurrently while it is full. In contrast, Rule (5) specifies that the bowl becomes empty if exactly one of the two actions is executed, regardless of its previous state. Finally, rules (6) and (7) provide the standard formalization of the *law of inertia* for the fluent $empty$ using strong negation.

To demonstrate that the equilibrium models of our formalization capture the intended behavior, we examine several representative traces. We begin with a labeled HT-trace $\mathbf{M} = (\langle\mathbf{T}, \mathbf{T}\rangle, \aleph)$ of length $\lambda = 1$ where $\mathbf{T} = (T_0) = (\{{\sim}empty\})$ and $\aleph(ll) = \emptyset, \aleph(lr) = \emptyset$. Evidently, $\mathbf{M}$ satisfies the initial condition (3) as ${\sim}empty \in T_0$. Furthermore, $\mathbf{M}$ satisfies the inertia laws (6) and (7) at 0, since both $\widehat{\circ}empty$ and $\widehat{\circ}{\sim}empty$ are vacuously true at the end of any trace. Similarly, the effect laws (4) and (5)

[2] This differs from a branching time setting, where it only requires one execution path for ρ.

[3] An executability law for an action formula α of form $\varphi \rightarrow \langle\alpha\rangle\,\top$ seems too strong in our linear semantics, since it imposes the execution of α rather then merely asserting its possibility as in a branching semantics.

hold because $[(ll \mathbin{\dot\wedge} \dot\neg lr) \mathbin{\dot\vee} (lr \mathbin{\dot\wedge} \dot\neg ll)]\, empty$ and $[ll \mathbin{\dot\wedge} lr] \sim empty$ are also trivially true at the last state. One can readily verify that $\mathbf{M}$ constitutes an equilibrium model of the theory (3)–(7).

Next, consider a trace $\langle (T_0, T_1), (T_0, T_1)\rangle$ of length $\lambda = 2$, where

$$(T_0, T_1) = (\{\sim empty\}, \{empty\}) \quad \text{and} \quad \aleph(ll) = \{(0,1)\} \quad \aleph(lr) = \emptyset.$$

Since $(0,1) \in \aleph(ll)$ and $(0,1) \notin \aleph(lr)$, the rule in (5) is triggered at 0, yielding $empty$ at 1. In contrast, the rule in (4) remains inapplicable as $(0,1) \notin \aleph(lr)$. Furthermore, the inertia law for $\sim empty$ in (7) is inapplicable, causing $\sim empty$ to be false at 1.

If we consider a trace of length $\lambda = 2$ where both hands are used,

$$(T_0, T_1) = (\{\sim empty\}, \{\sim empty\}) \quad \text{and} \quad \aleph(ll) = \{(0,1)\} \quad \aleph(lr) = \{(0,1)\},$$

the rule in (4) is triggered, ensuring $\sim empty \in T_1$. In this case, Rule (5) is inapplicable since $(0,1) \notin \|(ll \mathbin{\dot\wedge} \dot\neg lr) \mathbin{\dot\vee} (lr \mathbin{\dot\wedge} \dot\neg ll)\|^{\mathbf{M}}_{\models}$. Similarly, the inertia rules in (6)–(7) are not fired, and $empty$ remains false at 1.

Finally, in a trace where no action occurs between the initial and final state,

$$(T_0, T_1) = (\{\sim empty\}, \{\sim empty\}) \quad \text{and} \quad \aleph(ll) = \emptyset \quad \aleph(lr) = \emptyset,$$

neither effect law (4) nor (5) is triggered. Consequently, the inertia law in (7) ensures that $\sim empty$ persists at 1.

Our second example adapts the glass breaking scenario from [29] to incorporate concurrency. *A wall is struck with two hammers, h_1 and h_2, used either individually or simultaneously. Only simultaneous strikes may cause the wall to break, though this outcome is not guaranteed; multiple hits might be required before the wall finally breaks.* We formalize this scenario using the actions h_1, h_2, representing strikes with the respective hammers, and the fluent $broken$.

The following theory describes the resulting formalization in DEL^{af}:

$$\sim broken \tag{8}$$

$$\Box(\top \to [h_1 \mathbin{\dot\wedge} h_2]\,(broken \vee \neg broken)) \tag{9}$$

$$\Box(broken \to \widehat{\circ} broken) \tag{10}$$

$$\Box(\sim broken \wedge \neg\circ\, broken \to \widehat{\circ} \sim broken) \tag{11}$$

$$[(\sim broken?; (h_1 + h_2))^*]\, \neg broken \to \bot \tag{12}$$

The fact in (8) specifies that the wall is initially intact. Rule (9) states that striking the wall with both hammers concurrently may result in its breaking, leaving the truth value of $broken$ undetermined in the subsequent state. The inertia laws (10) and (11) ensure that a broken wall remains broken, whereas an intact wall remains so unless evidence of the contrary is provided. Lastly, the liveness property in (12) forbids equilibrium models in which the wall never breaks after any sequence of strikes. This rule, which is equivalent to $\neg\neg\langle(\sim broken?; (h_1 + h_2))^*\rangle\, broken$, effectively implements an 'until' loop and plays a role similar to that of dynamic constraints in *golog* [14].

While we omit the technical details, the reduction of DEL^{af} to DEL described in Section 4 enables the generation of the DFA in Figure 1, which illustrates the resulting transition system. With it, we can check that the labeled HT-trace $(\langle \mathbf{T}, \mathbf{T}\rangle, \aleph)$ is an equilibrium model of (8)–(12), where

$$\mathbf{T} = (\{\sim broken\}, \{\sim broken\}, \{broken\}, \{broken\}) \text{ and } \aleph(h_1) = \{(0,1),(1,2)\}, \aleph(h_2) = \{(1,2)\}.$$

3.2. Action Language $\mathcal{BC}$

To provide further evidence of the generality of our approach, we give next an embedding of action language $\mathcal{BC}$ [2, 30], as it encompasses the two primary approaches, $\mathcal{B}$ [1] and $\mathcal{C}$ [31]. An action description in $\mathcal{BC}$ is also based on a two-sorted alphabet, consisting of action and multi-valued state

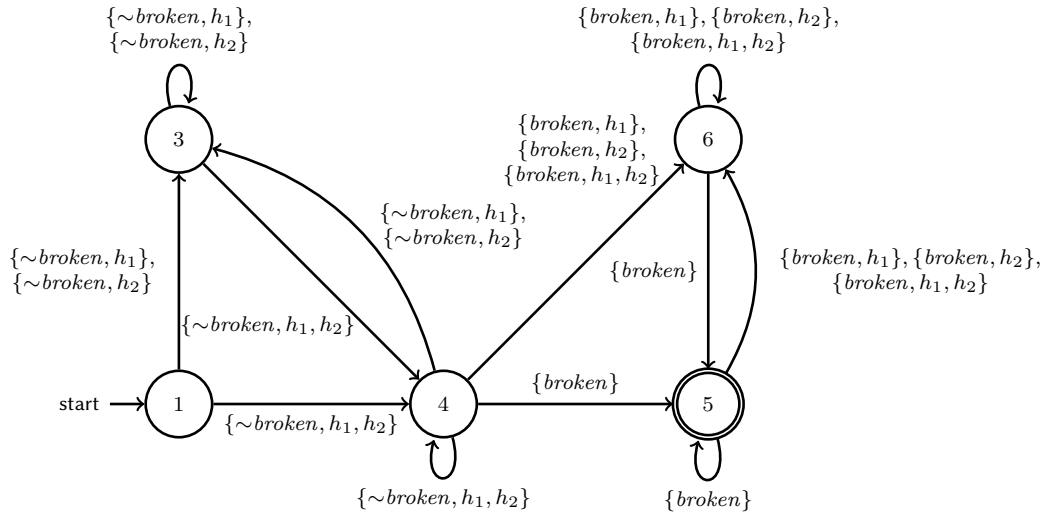


Figure 1: DFA accepting the equilibrium models of breaking a glass example. For each transition, all possible sets of fluents and actions that must hold are displayed. Every atom not displayed is considered as false.

variables (or fluents). In what follows, we restrict ourselves to two-valued fluents over 0 and 1, and abbreviate fluent atoms $f = 1$ and $f = 0$ by f and $\sim f$ to ease notation.[4]

An action description in $\mathcal{BC}$ consists of a finite set of static and dynamic laws. A *static law* is an expression of the form

$$f_0 \textbf{ if } f_1, \ldots, f_m \textbf{ ifcons } f_{m+1}, \ldots, f_n \tag{13}$$

where each f_i is a fluent atom for $0 \leq i \leq n$. A *dynamic law* is an expression of the form

$$f_0 \textbf{ after } a_1, \ldots, a_k, f_1, \ldots, f_m \textbf{ ifcons } f_{m+1}, \ldots, f_n \tag{14}$$

where each f_i is a fluent atom for $0 \leq i \leq n$ and each a_j is an action variable for $0 \leq j \leq k$. The keyword **ifcons** stands for "if consistent", while **if** and **after** have their literal meaning. Static laws describe causal dependencies between fluents in the same state, while dynamic laws describe direct effects of actions, when passing from one state to another. Accordingly, $\mathcal{BC}$ distinguishes statically determined fluents, which are permitted only in the heads of static laws, not dynamic laws.

We can translate $\mathcal{BC}$ into DEL^{af} as follows. Given an action description D, the theory $\Gamma(D)$ contains

1. for each static law from D of form (13), the formula

$$\Box(f_1 \wedge \cdots \wedge f_m \wedge \neg\sim f_{m+1} \wedge \cdots \wedge \neg\sim f_n \rightarrow f_0) \tag{15}$$

2. for each dynamic law from D of form (14), the formula

$$\Box(f_1 \wedge \cdots \wedge f_m \wedge \neg\circ\sim f_{m+1} \wedge \cdots \wedge \neg\circ\sim f_n \rightarrow [a_1 \dot{\wedge} \ldots \dot{\wedge} a_k]\, f_0) \tag{16}$$

3. for each non-statically determined fluent f the formula $(f \vee \sim f)$
4. for each fluent atoms f the following formulas

$$\Box\neg\neg(f \vee \sim f) \tag{17}$$

$$\Box(\neg(f \wedge \sim f)) \tag{18}$$

As regards semantics, [30] associates an action description D with a family of logic programs with strong negation $(P_l(D))_{l\geq 0}$ such that the stable models of $P_l(D)$ represent paths of length l in the

[4]Note that we overload the strong negation operator here.

transition system corresponding to D. The signature of each $P_l(D)$ consists of literals of form $f(i)$, $\sim f(i)$, and $a(i)$, for each fluent f and action variable a for $0 \leq i \leq l$.

For relating $\mathcal{BC}$ and DEL^{af}, we establish a one-to-one correspondence between the HT interpretations of a program $P_l(D)$ (cf. [21]) and the labeled HT traces of $\Gamma(D)$ of length $l+1$: Given an HT interpretation $\langle H, T\rangle$ of $P_l(D)$, we define a labeled HT-trace $((\langle H_i, T_i\rangle)_{i\in[0..l+1)}, \aleph)$ as

$$H_i = \{f \mid f(i) \in H\} \cup \{\sim f \mid \sim f(i) \in H\} \tag{19}$$
$$T_i = \{f \mid f(i) \in T\} \cup \{\sim f \mid \sim f(i) \in T\} \tag{20}$$
$$\aleph(a) = \{(i, i+1) \mid i \in [0..l), a(i) \in H\} \tag{21}$$

for $0 \leq i \leq l$. Conversely, given a labeled HT-trace $((\langle H_i, T_i\rangle)_{i\in[0..\lambda)}, \aleph)$ of length λ, we define the HT-interpretation $\langle H, T\rangle$ as

$$H = \{f(i), \sim g(i) \mid f, \sim g \in H_i, i \in [0..\lambda)\} \cup \{a(i) \mid (i, i+1) \in \aleph(a), i \in [0..\lambda-1)\} \tag{22}$$
$$T = \{f(i), \sim g(i) \mid f, \sim g \in T_i, i \in [0..\lambda)\} \cup \{a(i) \mid (i, i+1) \in \aleph(a), i \in [0..\lambda-1)\} \tag{23}$$

for $0 \leq i \leq l$. With this one-to-one correspondence, we obtain the following results.

Proposition 9. *Let $(\langle \mathbf{H}, \mathbf{T}\rangle, \aleph)$ be a labeled HT-trace of length $\lambda > 0$ and let $\langle H, T\rangle$ be an HT-interpretation being in the above correspondence. For any domain description D, we have*

$$\langle H, T\rangle \models P_{\lambda-1}(D) \text{ iff } (\langle \mathbf{H}, \mathbf{T}\rangle, \aleph), 0 \models \Gamma(D)$$
$$\langle H, T\rangle \dashv P_{\lambda-1}(D) \text{ iff } (\langle \mathbf{H}, \mathbf{T}\rangle, \aleph), 0 \dashv \Gamma(D)$$

Corollary 2. *Let $(\langle \mathbf{T}, \mathbf{T}\rangle, \aleph)$ be a labeled HT-trace of length $\lambda > 0$ and let $\langle T, T\rangle$ be an HT-interpretation being in the above correspondence. For any domain description D, $\langle T, T\rangle$ is an equilibrium model of $P_{\lambda-1}(D)$ as defined in [21] iff $(\langle \mathbf{T}, \mathbf{T}\rangle, \aleph)$ is an equilibrium model of $\Gamma(D)$.*

4. Reducing DEL^{af} to DEL$^{\sim}$

Despite its expressiveness for reasoning about action and change, DEL^{af} can be reduced to the simpler, one-sorted Dynamic Equilibrium Logic (DEL; [10, 11]), whose logic programming fragment is implemented by *telingo* [22]. More precisely, we consider DEL$^{\sim}$, the extension of DEL with strict negation defined in Appendix A.

The syntax of DHT^{af} rests upon a two-sorted alphabet $(\mathcal{A}, \mathcal{F})$ of strictly disjoint actions and fluents that cannot be used interchangeably. In contrast, DHT (and consequently DHT$^{\sim}$ and DEL$^{\sim}$) features only a single generic action, τ. To translate formulas from DHT^{af} into DHT$^{\sim}$, we drop this two-sorted distinction and treat all variables uniformly as fluents. In our terminology, this amounts to transforming formulas over the alphabet $(\mathcal{A}, \mathcal{F})$ into formulas over the (one-sorted) alphabet $(\{\tau\}, \mathcal{A} \cup \mathcal{F})$.

Given a state formula φ over $(\mathcal{A}, \mathcal{F})$, we define $\tau(\varphi)$ as the result of

1. replacing all operators $\dot{\top}, \dot{\neg}, \dot{\wedge}$ with $\top, \neg, \wedge$, respectively,
2. replacing each maximal action formula α with the path expression $\alpha?;\tau$, and
3. applying τ homomorphically to all remaining operators.

As this translation modifies only action formulas, $\tau(\varphi)$ is a state formula over $(\{\tau\}, \mathcal{A} \cup \mathcal{F})$.

For example, applying τ to the effect law in (4) yields the formula

$$\Box\,(\sim\!empty \rightarrow [ll \wedge lr?;\tau]\sim\!empty)\,.$$

This can be transformed into the formula $\Box\,((\sim\!empty \wedge ll \wedge lr) \rightarrow \widehat{\circ}\sim\!empty)$.

To establish a model correspondence, we first identify a specific class of HT-traces, termed actionable, that yields a bijection between labeled HT-traces in DHT^{af} and actionable HT-traces in DHT$^{\sim}$: An HT-trace $(\langle H_i, T_i\rangle)_{i\in[0..\lambda)}$ over $\widetilde{\mathcal{A}} \cup \widetilde{\mathcal{F}}$ of length λ is *actionable* if $(\langle H_i, T_i\rangle)_{i\in[0..\lambda)}$ is $\mathcal{A}$-complete and $H_{\lambda-1} \cap \widetilde{\mathcal{A}} = \{\sim a \mid a \in \mathcal{A}\}$. This property can be enforced as follows.

Proposition 10. *An HT-trace* $\mathbf{M}$ *of length* λ *over* $\widetilde{\mathcal{A}} \cup \widetilde{\mathcal{F}}$ *is actionable iff*

$$\mathbf{M}, 0 \models \{\Box(\neg\mathbb{F} \rightarrow (a \vee {\sim}a)), \Box(\mathbb{F} \rightarrow {\sim}a) \mid a \in \mathcal{A}\} \tag{24}$$

We now establish the bijective model correspondence. Given a labeled trace $(\langle H_i, T_i\rangle)_{i\in[0..\lambda)}, \aleph)$ of length λ over $(\mathcal{A}, \widetilde{\mathcal{F}})$, we define the HT-trace $(\langle H'_i, T'_i\rangle)_{i\in[0..\lambda)}$ of length λ over $\widetilde{\mathcal{A}} \cup \widetilde{\mathcal{F}}$, where for all $i \in [0..\lambda)$

$$\begin{aligned} H'_i &= H_i \cup \{a \mid (i, i+1) \in \aleph(a)\} \cup \{{\sim}a \mid (i, i+1) \notin \aleph(a)\} \text{ and} \\ T'_i &= T_i \cup \{a \mid (i, i+1) \in \aleph(a)\} \cup \{{\sim}a \mid (i, i+1) \notin \aleph(a)\}. \end{aligned}$$

Note that the obtained HT-trace is actionable.

Conversely, given an actionable trace $(\langle H'_i, T'_i\rangle)_{i\in[0..\lambda)}$ of length λ over $\widetilde{\mathcal{A}} \cup \widetilde{\mathcal{F}}$, we define the labeled trace $(\langle H_i, T_i\rangle)_{i\in[0..\lambda)}, \aleph)$ of length λ over $(\mathcal{A}, \widetilde{\mathcal{F}})$, where for all $i \in [0..\lambda)$

$$H_i = H'_i \cap \widetilde{\mathcal{F}}, \qquad T_i = T'_i \cap \widetilde{\mathcal{F}}, \text{ and} \qquad \aleph(a) = \{(i, i+1) \mid a \in H'_i \cap \mathcal{A}\}.$$

We obtain the following result.

Proposition 11. *Let* $\mathbf{M}$ *be a labeled HT-trace of length* λ *over* $(\mathcal{A}, \widetilde{\mathcal{F}})$ *and* $\mathbf{M}'$ *be an actionable HT-trace of length* λ *over* $\widetilde{\mathcal{A}} \cup \widetilde{\mathcal{F}}$ *satisfying the above correspondence.*

For every state formula φ *over* $(\mathcal{A}, \mathcal{F})$*, we have* $\mathbf{M}, k \models \varphi$ *iff* $\mathbf{M}', k \models \tau(\varphi)$ *for all* $k \in [0..\lambda)$*.*

Corollary 3. *Let* $\mathbf{M}$ *be a labeled HT-trace over* $(\mathcal{A}, \widetilde{\mathcal{F}})$ *and* $\mathbf{M}'$ *be an actionable HT-trace over* $\widetilde{\mathcal{A}} \cup \widetilde{\mathcal{F}}$ *satisfying the above correspondence.*

For every formula φ *over* $(\mathcal{A}, \mathcal{F})$*, we have that* $\mathbf{M}$ *is an equilibrium model of* φ *in DEL*^{af} *iff* $\mathbf{M}'$ *is an equilibrium model of* $\tau(\varphi)$ *in DEL*$^{\sim}$ *(as defined in Appendix A).*

5. Related work

Our framework builds upon the foundations of propositional Dynamic Logic (DL; [7]). While the latter is generally interpreted over Kripke frames, our formalism adopts a linear-time semantics over finite traces. Consequently, formulas receive a different semantics: for example, what expresses executability in DL corresponds to actual execution in our setting. Nevertheless, DEL^{af} preserves the syntactic structure of DL, relying on the interplay between path expressions and state formulas. Early work on the interpretation of DL over linear traces was done in [15, 17]. In contrast to DEL^{af}, this approach is monotonic and permits only a single action per transition. Accordingly, it has no direct means to express the parallel execution of actions. More recent work in this direction resulted in a variant based upon linear finite traces [16], which retains a DL-style syntax yet in a one-sorted setting.

Rosenschein [12] pioneered the use of DL for planning. Extensions of DL that incorporate concurrent path expressions [32, 33] typically rely on accessibility relations defined over sets of states. In contrast, our approach is limited to the parallel execution of action formulas. DL has also inspired high-level action languages such as *golog* [14] and *congolog* [34]. These formalisms decouple the specification of action effects from the procedural constraints that govern admissible courses of action. This modeling philosophy resonates with our approach, which leverages path expressions to restrict valid state trajectories. However, unlike DEL^{af}, these frameworks are inherently monotonic and consequently rely on monotonic solutions to the frame problem. More broadly, DL has served as a foundation for meta-theoretical investigations of action formalisms, encompassing analyses of consistency, ramifications, and qualification [35], as well as modularity and the revision of domain descriptions [36, 37]. Furthermore, comparative studies have employed DL as a unifying semantic framework for relating various monotonic and non-monotonic solutions to the frame problem [38]. Notably, the non-monotonic nature of DEL^{af} enables a succinct representation of inertia, providing an effective solution to the frame problem.

Non-monotonic variants of DL were introduced in [39, 10, 29] as extensions of Equilibrium Logic [21],[5] analogous to its linear temporal extension [41, 5]. Our approach advances these logics by introducing a two-sorted setting, effectively establishing a clear distinction between action and state formulas. A reduct-based non-monotonic extension of DL was proposed in [4], building upon the aforementioned linear-time variant [15]. This approach was later reconstructed as an extension of Equilibrium Logic in [39]. While it also relies on a two-sorted alphabet, it is restricted to single-action transitions and omits test operations in path expressions.

Finally, action languages [27, 1, 42, 2] also strictly separate actions and fluents, typically defining their semantics over transition systems. These formalisms provide concise means for representing action and change, and admit concurrency [27]. They generally resolve the frame problem through built-in minimal-change assumptions, which are often compiled into stable-model semantics [2, 30]. Notably, the resulting inertia laws closely resemble the formulations employed in our approach.

6. Conclusion

We presented a formal framework for reasoning about action and change that relies on Dynamic Logic. Our approach unifies the core two-sorted language of [8, 9] with the non-monotonic semantic foundations introduced in [10, 11]. Our primary objective was not merely to introduce yet another standalone formalism for action and change, but rather to bridge the gap between high-level knowledge representation and practical computability. By synthesizing established insights from the literature, we formulated a system that is highly expressive by accommodating rich temporal constraints and explicitly distinguishing between action and state variables, while remaining easily reducible to a single-sorted structure. Consequently, rather than defining a purely theoretical logic from scratch, our framework is explicitly designed to be amenable to implementation on top of state-of-the-art ASP systems. Specifically, our reduction paves the way for evaluating these expressive dynamic models using the temporal ASP solver *telingo*, leveraging the robust computational infrastructure of the ASP system *clingo*.

7. Declaration on Generative AI

During the preparation of this work, the authors used ChatGPT in order to: Grammar and spelling check, Paraphrase and reword. After using this tool/service, the authors reviewed and edited the content as needed and take full responsibility for the publication's content.

Acknowledgments

This work has been partially supported by DFG grant SCHA 550/15.

References

[1] M. Gelfond, V. Lifschitz, Action languages, Electronic Transactions on Artificial Intelligence 3 (1998) 193–210. URL: http://www.ep.liu.se/ej/etai/1998/007/.

[2] J. Lee, V. Lifschitz, F. Yang, Action language BC: Preliminary report, in: [43], 2013, pp. 983–989.

[3] P. Cabalar, G. Pérez Vega, Temporal equilibrium logic: A first approach, in: R. Moreno-Díaz, F. Pichler, A. Quesada-Arencibia (Eds.), Proceedings of the Eleventh International Conference on Computer Aided Systems Theory (EUROCAST'17), volume 4739 of *Lecture Notes in Computer Science*, Springer-Verlag, 2007, pp. 241–248.

[5] Notably, [29] employs an alternative form of strong negation aligned with reduct-based semantics, whereas our framework adopts the N5-based operator [40] originally used for extending Equilibrium Logic [21].

[4] L. Giordano, A. Martelli, D. Theseider Dupré, Reasoning about actions with temporal answer sets, Theory and Practice of Logic Programming 13 (2013) 201–225.
[5] F. Aguado, P. Cabalar, M. Diéguez, G. Pérez, T. Schaub, A. Schuhmann, C. Vidal, Linear-time temporal answer set programming, Theory and Practice of Logic Programming 23 (2023) 2–56. doi:10.1017/S1471068421000557.
[6] V. Lifschitz, Answer set planning, in: D. de Schreye (Ed.), Proceedings of the International Conference on Logic Programming (ICLP'99), MIT Press, 1999, pp. 23–37.
[7] D. Harel, J. Tiuryn, D. Kozen, Dynamic Logic, MIT Press, 2000. doi:10.1145/568438.568456.
[8] G. De Giacomo, M. Lenzerini, Enhanced propositional dynamic logic for reasoning about concurrent actions, in: C. Boutilier, M. Goldszmidt, T. Dean, S. Hanks, D. Heckerman, R. Reiter (Eds.), Working notes of the AAAI Spring Symposium on Extending Theories of Action: Formal and Practical Applications, 1995, pp. 62–67.
[9] G. De Giacomo, M. Lenzerini, PDL-based framework for reasoning about actions, in: M. Gori, G. Soda (Eds.), Proceedings of the fourth Congress of the Italian Association for Artificial Intelligence (AIIA'95), volume 992 of *Lecture Notes in Computer Science*, Springer-Verlag, 1995, pp. 103–114. doi:10.1007/3-540-60437-5_10.
[10] A. Bosser, P. Cabalar, M. Diéguez, T. Schaub, Introducing temporal stable models for linear dynamic logic, in: M. Thielscher, F. Toni, F. Wolter (Eds.), Proceedings of the Sixteenth International Conference on Principles of Knowledge Representation and Reasoning (KR'18), AAAI Press, 2018, pp. 12–21.
[11] P. Cabalar, M. Diéguez, T. Schaub, Towards dynamic answer set programming over finite traces, in: [44], 2019, pp. 148–162. doi:10.1007/978-3-030-20528-7_12.
[12] S. Rosenschein, Plan synthesis: A logical perspective, in: P. Hayes (Ed.), Proceedings of the Seventh International Joint Conference on Artificial Intelligence (IJCAI'81), William Kaufmann, 1981, pp. 331–337. URL: http://ijcai.org/Proceedings/81-1/Papers/061.pdf.
[13] A. Herzig, I. Varzinczak, Domain descriptions should be modular, in: R. López de Mántaras, L. Saitta (Eds.), Proceedings of the Sixteenth Eureopean Conference on Artificial Intelligence (ECAI'04), IOS Press, 2004, pp. 348–352.
[14] H. Levesque, R. Reiter, Y. Lespérance, F. Lin, R. Scherl, GOLOG: A logic programming language for dynamic domains., Journal of Logic Programming 31 (1997) 59–83. doi:10.1016/S0743-1066(96)00121-5.
[15] J. Henriksen, P. Thiagarajan, Dynamic linear time temporal logic, Annals of Pure and Applied Logic 96 (1999) 187–207.
[16] G. De Giacomo, M. Vardi, Linear temporal logic and linear dynamic logic on finite traces, in: [43], 2013, pp. 854–860.
[17] L. Giordano, A. Martelli, C. Schwind, Reasoning about actions in dynamic linear time temporal logic, Logic Journal of the IGPL 9 (2001) 273–288.
[18] M. Gelfond, V. Lifschitz, The stable model semantics for logic programming, in: R. Kowalski, K. Bowen (Eds.), Proceedings of the Fifth International Conference and Symposium of Logic Programming (ICLP'88), MIT Press, 1988, pp. 1070–1080. doi:10.1201/b10397-6.
[19] D. Pearce, A new logical characterisation of stable models and answer sets, in: J. Dix, L. Pereira, T. Przymusinski (Eds.), Proceedings of the Sixth International Workshop on Non-Monotonic Extensions of Logic Programming (NMELP'96), volume 1216 of *Lecture Notes in Computer Science*, Springer-Verlag, 1997, pp. 57–70. doi:10.1007/BFb0023801.
[20] A. Heyting, Die formalen Regeln der intuitionistischen Logik, in: Sitzungsberichte der Preussischen Akademie der Wissenschaften, Deutsche Akademie der Wissenschaften zu Berlin, 1930, pp. 42–56.
[21] D. Pearce, Equilibrium logic, Annals of Mathematics and Artificial Intelligence 47 (2006) 3–41. doi:10.1007/s10472-006-9028-z.
[22] P. Cabalar, R. Kaminski, P. Morkisch, T. Schaub, telingo = ASP + time, in: [44], 2019, pp. 256–269. doi:10.1007/978-3-030-20528-7_19.
[23] F. Aguado, P. Cabalar, M. Diéguez, G. Pérez, C. Vidal, Temporal equilibrium logic with past operators, Journal of Applied Non-Classical Logics 27 (2017) 161–177.

[24] M. Ortiz, M. Osorio, Strong negation and equivalence in the safe belief semantics, Journal of Logic and Computation 17 (2007) 499–515. doi:10.1093/logcom/exm011.
[25] F. Aguado, P. Cabalar, M. Diéguez, G. Pérez, C. Vidal, Temporal equilibrium logic: a survey, Journal of Applied Non-Classical Logics 23 (2013) 2–24. doi:10.1080/11663081.2013.798985.
[26] D. Vakarelov, Notes on n-lattices and constructive logic with strong negation, Studia Logica 36 (1977) 109–125.
[27] C. Baral, M. Gelfond, Representing concurrent actions in extended logic programming, in: Proceedings of the Thirteenth International Joint Conference on Artificial Intelligence (IJCAI'93), volume 2, Morgan Kaufmann Publishers, 1993, p. 866–871.
[28] C. Baral, M. Gelfond, Reasoning about effects of concurrent actions, Journal of Logic Programming 31 (1997) 85–117. doi:10.1016/S0743-1066(96)00140-9.
[29] F. Aguado, P. Cabalar, J. Fandinno, G. Pérez, C. Vidal, Explicit negation in linear-dynamic equilibrium logic, in: G. De Giacomo, A. Catalá, B. Dilkina, M. Milano, S. Barro, A. Bugarín, J. Lang (Eds.), Proceedings of the Twenty-fourth European Conference on Artificial Intelligence (ECAI'20), IOS Press, 2020, pp. 569–576.
[30] F. Yang, Representing Actions in Logic-Based Languages, Ph.D. thesis, University of Texas at Austin, 2014. URL: https://www.cs.utexas.edu/~vl/dissertations/yang.pdf.
[31] E. Giunchiglia, J. Lee, V. Lifschitz, N. McCain, H. Turner, Nonmonotonic causal theories, Artificial Intelligence 153 (2004) 49–104.
[32] D. Peleg, Concurrent dynamic logic, Journal of the ACM 34 (1987) 450–479. URL: https://doi.org/10.1145/23005.23008. doi:10.1145/23005.23008.
[33] R. Goldblatt, Parallel action: Concurrent dynamic logic with independent modalities, Studia Logica 51 (1992) 551–578. doi:10.1007/bf01028975.
[34] G. De Giacomo, Y. Lespérance, H. Levesque, ConGolog, a concurrent programming language based on the situation calculus, Artificial Intelligence 121 (2000) 109–169.
[35] D. Zhang, S. Chopra, N. Foo, Consistency of action descriptions, in: M. Ishizuka, A. Sattar (Eds.), Proceedings of the Seventh Pacific Rim International Conference on Artificial Intelligence (PRICAI'02), volume 2417 of *Lecture Notes in Computer Science*, Springer-Verlag, 2002, pp. 70–79.
[36] A. Herzig, I. Varzinczak, Domain descriptions should be modular, in: J. Delgrande, T. Schaub (Eds.), Proceedings of the Tenth International Workshop on Nonmonotonic Reasoning (NMR'04), 2004, pp. 204–211.
[37] I. Varzinczak, What Is a Good Domain Description? Evaluating & Revising Action Theories in Dynamic Logic, Ph.D. thesis, Paul Sabatier University, 2006. URL: https://tel.archives-ouvertes.fr/tel-00319220.
[38] D. Zhang, N. Foo, Frame problem in dynamic logic, Journal of Applied Non-Classical Logics 15 (2005) 215–239. doi:10.3166/jancl.15.215-239.
[39] F. Aguado, G. Pérez, C. Vidal, Integrating temporal extensions of answer set programming, in: P. Cabalar, T. Son (Eds.), Proceedings of the Twelfth International Conference on Logic Programming and Nonmonotonic Reasoning (LPNMR'13), volume 8148 of *Lecture Notes in Artificial Intelligence*, Springer-Verlag, 2013, pp. 23–35. doi:10.1007/978-3-642-40564-8_3.
[40] D. Nelson, Constructible falsity, The Journal of Symbolic Logic 14 (1949) 16–26. doi:10.2307/2268973.
[41] P. Cabalar, G. Pérez Vega, Temporal equilibrium logic: A first approach, in: R. Moreno-Díaz, F. Pichler, A. Quesada-Arencibia (Eds.), Revised Selected Papers of the Eleventh International Conference on Computer Aided Systems Theory (EUROCAST'07), volume 4739 of *Lecture Notes in Computer Science*, Springer-Verlag, 2007, pp. 241–248.
[42] T. Eiter, W. Faber, N. Leone, G. Pfeifer, A. Polleres, A logic programming approach to knowledge-state planning, Artificial Intelligence 144 (2003) 157–211.
[43] F. Rossi (Ed.), Proceedings of the Twenty-third International Joint Conference on Artificial Intelligence (IJCAI'13), IJCAI/AAAI Press, 2013.
[44] M. Balduccini, Y. Lierler, S. Woltran (Eds.), Proceedings of the Fifteenth International Conference on Logic Programming and Nonmonotonic Reasoning (LPNMR'19), volume 11481 of *Lecture Notes*

in Artificial Intelligence, Springer-Verlag, 2019.
[45] F. Aguado, P. Cabalar, J. Fandinno, D. Pearce, G. Pérez, C. Vidal, Revisiting explicit negation in answer set programming, Theory and Practice of Logic Programming 19 (2019) 908–924.
[46] P. Ferraris, Logic programs with propositional connectives and aggregates, ACM Transactions on Computational Logic 12 (2011) 25:1–25:40. doi:10.1145/1970398.1970401.

A. Definition of logics DHT$^\sim$ and DEL$^\sim$

Given a set $\mathcal{F}$ of propositional (fluent) variables, *dynamic formulas* φ and *path expressions* ρ are mutually defined as follows:

$$\varphi ::= f \in \mathcal{F} \mid \bot \mid {\sim}\varphi \mid \langle\rho\rangle\,\varphi \mid [\rho]\,\varphi \tag{25}$$
$$\rho ::= \tau \mid \rho + \rho \mid \rho;\rho \mid \rho^* \mid \varphi? \mid \rho^- \tag{26}$$

The semantics of dynamic formulas is defined in terms of *traces* over $\widetilde{\mathcal{F}}$, as described in Section 2.2.

An HT-trace $\mathbf{M} = \langle \mathbf{H}, \mathbf{T}\rangle$ of length λ over $\widetilde{\mathcal{F}}$ satisfies (resp. falsifies) a dynamic formula φ over $\mathcal{F}$ at $i \in [0..\lambda)$, in symbols $\mathbf{M}, i \models \varphi$ (resp. $\mathbf{M}, i \dashv \varphi$), if the following conditions hold:

1. $\mathbf{M}, i \models f$ if $f \in H_i$ for $f \in \mathcal{F}$
2. $\mathbf{M}, i \not\models \bot$
3. $\mathbf{M}, i \models {\sim}\psi$ if $\mathbf{M}, i \dashv \psi$
4. $\mathbf{M}, i \models \langle\rho\rangle\,\varphi$ if $\mathbf{M}, j \models \varphi$ for some $(i,j) \in \|\rho\|^{\mathbf{M}}_{\models}$
5. $\mathbf{M}, i \models [\rho]\,\varphi$ if $\langle \mathbf{X}, \mathbf{T}\rangle, j \models \varphi$ for all $(i,j) \in \|\rho\|^{\langle \mathbf{X},\mathbf{T}\rangle}_{\models}$ and $\mathbf{X} \in \{\mathbf{H}, \mathbf{T}\}$
6. $\mathbf{M}, i \dashv f$ if ${\sim}f \in H_i$ for $f \in \mathcal{F}$
7. $\mathbf{M}, i \dashv \bot$
8. $\mathbf{M}, i \dashv {\sim}\psi$ if $\mathbf{M}, i \models \psi$
9. $\mathbf{M}, i \dashv \langle\rho\rangle\,\varphi$ if $\mathbf{M}, j \dashv \varphi$ for all $(i,j) \in \|\rho\|^{\mathbf{M}}_{\not\models}$
10. $\mathbf{M}, i \dashv [\rho]\,\varphi$ if $(\langle \mathbf{H}, \mathbf{T}\rangle, \aleph), j \dashv \varphi$ for some $(i,j) \in \|\rho\|^{\langle \mathbf{H},\mathbf{T}\rangle}_{\models}$

where the relations $\|\rho\|^{\mathbf{M}}_{\models} \subseteq [0..\lambda) \times [0..\lambda)$ and $\|\rho\|^{\mathbf{M}}_{\not\models} \subseteq [0..\lambda) \times [0..\lambda)$ are defined for $\tau \notin \mathcal{F}$, state formula φ, and path expressions ρ, ϱ, as

$$\|\tau\|^{\mathbf{M}}_{\models} \stackrel{\text{def}}{=} I_\lambda \qquad \|\tau\|^{\mathbf{M}}_{\not\models} \stackrel{\text{def}}{=} I_\lambda$$
$$\|\varphi?\|^{\mathbf{M}}_{\models} \stackrel{\text{def}}{=} \{(i,i) \mid \mathbf{M}, i \models \varphi\} \qquad \|\varphi?\|^{\mathbf{M}}_{\not\models} \stackrel{\text{def}}{=} \{(i,i) \mid \mathbf{M}, i \not\models \varphi\}$$
$$\|\rho^-\|^{\mathbf{M}}_{\models} \stackrel{\text{def}}{=} \{(k,i) \mid (i,k) \in \|\rho\|^{\mathbf{M}}_{\models}\} \qquad \|\rho^-\|^{\mathbf{M}}_{\not\models} \stackrel{\text{def}}{=} \{(k,i) \mid (i,k) \in \|\rho\|^{\mathbf{M}}_{\not\models}\}$$
$$\|\rho + \varrho\|^{\mathbf{M}}_{\models} \stackrel{\text{def}}{=} \|\rho\|^{\mathbf{M}}_{\models} \cup \|\varrho\|^{\mathbf{M}}_{\models} \qquad \|\rho + \varrho\|^{\mathbf{M}}_{\not\models} \stackrel{\text{def}}{=} \|\rho\|^{\mathbf{M}}_{\not\models} \cup \|\varrho\|^{\mathbf{M}}_{\not\models}$$
$$\|\rho\,;\,\varrho\|^{\mathbf{M}}_{\models} \stackrel{\text{def}}{=} \|\rho\|^{\mathbf{M}}_{\models} \circ \|\varrho\|^{\mathbf{M}}_{\models} \qquad \|\rho\,;\,\varrho\|^{\mathbf{M}}_{\not\models} \stackrel{\text{def}}{=} \|\rho\|^{\mathbf{M}}_{\not\models} \circ \|\varrho\|^{\mathbf{M}}_{\not\models}$$
$$\|\rho^*\|^{\mathbf{M}}_{\models} \stackrel{\text{def}}{=} \left(\|\rho\|^{\mathbf{M}}_{\models}\right)^* \qquad \|\rho^*\|^{\mathbf{M}}_{\not\models} \stackrel{\text{def}}{=} \left(\|\rho\|^{\mathbf{M}}_{\not\models}\right)^*$$

An HT-trace of form $\langle \mathbf{T}, \mathbf{T}\rangle$ is a *temporal equilibrium model* of a theory Γ if it is a model of Γ and there is no other model $\langle \mathbf{H}, \mathbf{T}\rangle$ of Γ with $\mathbf{H} < \mathbf{T}$.

Our treatment of strong negation in both DEL^{af} and DEL$^\sim$ builds upon Nelson's logic N5 [40], reflecting its original application in extending Equilibrium Logic [21]. This choice ensures consistency with Yang's HT-based characterization of the action language $\mathcal{BC}$ [30], which relies on the same concept of negation, thereby facilitating our embedding of $\mathcal{BC}$ into DEL^{af}.

A different form of strong negation for DEL was proposed in [29]. Its semantics is based on the alternative concept explored in [45], which aligns with the reduct-based semantics for nested expressions [46]. Regarding DHT$^\sim$, the primary technical distinction between the two approaches manifests in the dissatisfaction condition for the box modality: the alternative semantics employs the weaker relation $\|\rho\|^{\langle \mathbf{T},\mathbf{T}\rangle}_{\models}$ in Condition 10. instead of $\|\rho\|^{\langle \mathbf{H},\mathbf{T}\rangle}_{\models}$. The definition of DEL^{af} and its reduction to DEL could equally be based on this alternative formulation, especially since both concepts of implication are interdefinable [45].