On Temporal ASP with Eager Unfoldable Operators

Thomas Eiter , Davide Soldà
Institute of Logic and Computation
Vienna University of Technology (TU Wien), Vienna, Austria
{thomas.eiter, davide.solda}@tuwien.ac.at

Abstract

Temporal Equilibrium Logic (TEL) extends Answer Set Programming (ASP) with linear-time temporal operators (LTL), enabling reasoning about dynamic systems. However, TEL enforces strong minimization criteria that may preclude intuitive models. Liveness formulas, for instance, tend to fail to have infinite equilibrium models, as TEL minimization postpones satisfaction forever. We address this limitation by introducing eager temporal operators (eager Until, eager Release, etc.), and present non-disjunctive temporal programs (NDTP) as a framework for modeling dependencies, inertia, and non-determinism. The fragment of tight temporal programs (TTP), which can be recognized efficiently based on automata techniques for loop detections, guarantees polynomial encodability into LTL. Practical examples, such as request-grant protocols and user permissions in distributed systems, illustrate the applicability of our approach.

1 Introduction

Answer Set Programming (ASP) [Brewka *et al.*, 2011; Lifschitz, 2019] has been widely applied in dynamic environments, including applications in planning, multi-agent systems, and reasoning about evolving domains [Falkner *et al.*, 2018]. ASP semantics is particularly well-suited for such settings due to its capability of expressing transitive closure, addressing the frame problem (via inertia rules), and providing both default and strong negation, which enables rich and flexible modeling possibilities. Temporal Equilibrium Logic (TEL) [Aguado *et al.*, 2023] builds on this foundation by bridging the gap between the expressive power of linear-time temporal logic (LTL) [Pnueli, 1977] operators and the stable semantics of ASP. By integrating these paradigms, TEL provides a unified framework for reasoning about dynamic systems while preserving the non-monotonic reasoning capabilities and desirable properties of ASP.

A key distinction of TEL compared to LTL lies in its capability to address (i) non-determinism by naturally encoding various assumptions about unseen data, (ii) the inertia law, which allows for reasoning about persistence of facts over time, and (iii) dependencies, expressed in a rule-based manner, that are fundamental to many temporal systems.

Example 1 (Liveness property)**.** A user may issue a "request" for a resource in a system that must be granted at some future point in time. The rules governing this system dictate that a "grant" action g can only occur if a corresponding "request" r has been made in a prior step, in formulas $\Box(r \rightarrow \Diamond g)$. Classical LTL semantics cannot fully capture this dependency, i.e., a grant g can appear even without a request r. To model it in LTL, one must also include $\Box(g \rightarrow \blacklozenge r)$, ensuring that every grant g occurs due to a request r. However, this creates a cascading problem: if requests r themselves must be justified by other temporal rules, complicating the modeling, cf. *frame problem* as discussed in [McCarthy and Hayes, 1981].

While TEL effectively captures these temporal dependencies, its strong minimization condition often leads to unintended consequences. Specifically, if a request is derived infinitely often by a formula such as $\Box\Diamond r$, no infinite equilibrium model exists: after removing the first r from a model, we still have a model, which leads to an infinite chain of smaller models that compromises stability. For eventual persistence conditions such as $\Diamond\Box p$, the situation is similar.

To address this issue, we introduce a novel version of the *until*, *release*, *since*, and *trigger* operators, termed *eager* operators. These operators temper the strong minimization by suspending it until the operator is fulfilled along the trace. This yields new versions of derived operators, such as *eventually* and *once*, which align better with intuitive reading.

The proposed suspension ensures that in Example 1, we have infinite stable models of infinite length. For finitary semantics, the standard definition of *eventually* would in Example 1 place a grant g only in the very last state, from which it cannot be removed. However, such behavior does arguably not align with the intuitive reading of the formulas in Example 1. Therefore, even in the case of finite traces, we consider the eager version of the temporal operator a preferable choice to accurately capture the intended behavior.

Example 2 (Safety property)**.** For another scenario where dependencies matter, assume there are *admin* users and *regular* user. Admin users can promote a regular user to be an admin while regular users can't. Thus, a schematic rule such as $\Box(admin(X), promotes(X, Y), user(Y) \rightarrow admin(Y))$, where X and Y are placeholders for the grounded version of

the formula, should not self-support the predicate $admin$.

Our main contributions are summarized as follows.

• We introduce new versions of the temporal operators that unfold eagerly. While they are indifferent in LTL, they suspend minimization and thus allow for stable models in scenarios where such models are intuitively expected. Notably, the new versions are expressible in TEL and the resulting extension of TEL has the same complexity, i.e., TEL-satisfiability is EXPSPACE-complete [Bozzelli and Pearce, 2015].

• We define $NDTP$ as a rule-based fragment of the new language that, on the one hand, overcomes the problem of strong minimization and, on the other hand, has considerably lower complexity. The former is achieved by using eager temporal operators in rule heads and the latter by disallowing disjunction, which quickly leads to EXPSPACE-hardness [Šimkus, 2010]. We show that $NDTP$ programs have benign properties and that deciding TEL-satisfiability for them is in EXPTIME in general, while model checking is feasible in polynomial time.

• Based on $NDTP$, we define a tight version of temporal programs (TTP). By generalizing results for ordinary tight logic programs [Erdem and Lifschitz, 2003], we show that the stable models of TTP programs are obtained as the LTL-models of their temporal Clark's completion that we define. As a result, we obtain a polynomial encoding of TTP into LTL. Furthermore, we show that the tightness of temporal programs can be decided efficiently in NL, resorting to one-counter automata.

The use of $NDTP$ as a starting point can be seen as a temporal extension of normal programs (or non-disjunctive programs as in [Erdem and Lifschitz, 2003]), which is the standard fragment on top of which the notion of tightness is usually defined [Dodaro *et al.*, 2023; Lin and Zhao, 2003]. However, this approach has not yet been extended to ASP with modalities, which are highly useful logical operators for various domains. Our results enable bridging temporal nonmonotonicity with LTL-based model checkers such as `SPIN` [Holzmann, 1997], `nuXmv` [Cavada *et al.*, 2014], and `BLACK` [Geatti *et al.*, 2021].

Organization. The remainder of this paper is organized as follows. Section 2 provides a formal background on TEL. Section 3 introduces the new operators and shows their practical applicability. Section 4 introduces the $NDTP$ fragment, while Section 5 presents tight temporal programs. In Section 6, we discuss related work and conclude.

2 Preliminaries

Both TEL and THT [Aguado *et al.*, 2023] share the same syntax as LTL. Here we introduce the grammar

$$F ::= \top \mid \bot \mid p \mid F \vartriangle F \mid \circ F \mid \bullet F \mid F \mathsf{O} F \qquad (1)$$

where $p \in \mathcal{P}$ for a finite set $\mathcal{P}$ of propositional atoms, $\vartriangle \in \{\wedge, \vee, \rightarrow\}$, and $\mathsf{O} \in \{\mathbb{U}, \mathbb{R}, \mathbb{S}, \mathbb{T}\}$. Negation is defined as $\neg\phi := \phi \rightarrow \bot$. As usual, $\Box$ (globally) is defined by $\Box\phi := \bot\,\mathbb{R}\,\phi$; $\Diamond$ (eventually) by $\Diamond\phi := \top\,\mathbb{U}\,\phi$; $\blacksquare$ (historically) by $\blacksquare\phi := \bot\,\mathbb{T}\,\phi$; $\blacklozenge$ (once) by $\blacklozenge\phi := \top\,\mathbb{S}\,\phi$; $\hat{\circ}$ (weak next) by $\hat{\circ}\phi := \circ\phi \vee \neg\circ\top$; and $\hat{\bullet}$ (weak previous) by $\hat{\bullet}\phi := \bullet\phi \vee \neg\bullet\top$. Furthermore, for any unary operator u, we let $\times^0\phi$ denote ϕ and $\times^{i+1}\phi$ denote $\mathsf{u}\mathsf{u}^i\phi$, for $i \geq 0$.

The semantics of THT is defined via THT-traces (or simply traces, if clear from context), which are finite or infinite sequences $\langle \mathbf{H}, \mathbf{T} \rangle$ of pairs $\langle H_i, T_i \rangle$, where $H_i \subseteq T_i \subseteq \mathcal{P}$ for each $0 \leq i < \lambda$, where λ can be either in $\mathbb{N}$ or ω. Both $\mathbf{H}$ and $\mathbf{T}$ are traces as usual (LTL-traces), i.e., sequences $\mathbf{H} = H_0, H_1, \ldots$ resp. $\mathbf{T} = T_0, T_1, \ldots$ of sets of atoms. Given a THT-trace $\mathbf{I}$ (or an LTL-trace $\mathbf{T}$), we denote its length by $\lambda_{\mathbf{I}}$ (resp. $\lambda_{\mathbf{T}}$).

Definition 1 (THT-Satisfaction)**.** *Satisfaction of a THT formula by a THT-trace $\mathbf{I} = \langle \mathbf{H}, \mathbf{T} \rangle$ at time k, where $0 \leq k$ is integer, is inductively defined as follows:*

1. $\mathbf{I}, k \not\models \bot$ *and* $\mathbf{I}, k \not\models \top$
2. $\mathbf{I}, k \models p$ *if* $p \in H_k$*, for any atom* $p \in \mathcal{P}$
3. $\mathbf{I}, k \models \phi \vee \psi$ *if* $\mathbf{I}, k \models \phi$ *or* $\mathbf{I}, k \models \psi$
4. $\mathbf{I}, k \models \phi \wedge \psi$ *if* $\mathbf{I}, k \models \phi$ *and* $\mathbf{I}, k \models \psi$
5. $\mathbf{I}, k \models \phi \rightarrow \psi$ *if* $\begin{cases} \langle \mathbf{T}, \mathbf{T} \rangle, k \not\models \phi \text{ or } \langle \mathbf{T}, \mathbf{T} \rangle, k \models \psi, \text{ and} \\ \mathbf{I}, k \not\models \phi \text{ or } \mathbf{I}, k \models \psi \end{cases}$
6. $\mathbf{I}, k \models \circ\,\phi$ *if* $k + 1 < \lambda_{\mathbf{I}}$ *and* $\mathbf{I}, k + 1 \models \phi$
7. $\mathbf{I}, k \models \phi\,\mathbb{U}\,\psi$ *if there is* $j \geq k$ *s.t.* $\mathbf{I}, j \models \psi$*, and for all* $j' \in [k, j)$*,* $\mathbf{I}, j' \models \phi$
8. $\mathbf{I}, k \models \phi\,\mathbb{R}\,\psi$ *if for all* $j \geq k$ *s.t.* $\mathbf{I}, j \not\models \psi$*, there exists* $j' \in [k, j)$*,* $\mathbf{I}, j' \models \phi$
9. $\mathbf{I}, k \models \bullet\,\phi$ *if* $k > 0$ *and* $\mathbf{I}, k - 1 \models \phi$
10. $\mathbf{I}, k \models \phi\,\mathbb{S}\,\psi$ *if there is* $j \leq k$ *s.t.* $\mathbf{I}, j \models \psi$*, and for all* $j' \in (j, k)$*,* $\mathbf{I}, j' \models \phi$
11. $\mathbf{I}, k \models \phi\,\mathbb{T}\,\psi$ *if for all* $j \geq k$ *s.t.* $\mathbf{I}, j \not\models \psi$*, there exists* $j' \in [k, j)$*,* $\mathbf{I}, j' \models \phi$*,*

We recall that the persistence property holds for THT, i.e., that if a proposition is true in a here state H_i, it remains true in T_i, reflecting the notion that truth is preserved.

Proposition 1 (Persistence, [Aguado *et al.*, 2023])**.** *For any formula ϕ and THT-trace $\mathbf{I} = \langle \mathbf{H}, \mathbf{T} \rangle$, $\mathbf{I} \models \phi$ implies $\mathbf{T} \models \phi$.*

As a consequence, interpreting the negation $\neg\phi$ in $\langle \mathbf{H}, \mathbf{T} \rangle$ at time point i amounts to ϕ not holding in $\mathbf{T}$ at i. Formally,

Corollary 1. *For any formula ϕ and THT-trace $\mathbf{I} = \langle \mathbf{H}, \mathbf{T} \rangle$, $\mathbf{I} \models \neg\phi$ iff $\mathbf{T} \not\models \phi$.*

Furthermore, LTL and THT are strongly related, as $\langle \mathbf{T}, \mathbf{T} \rangle \models \phi$ iff $\mathbf{T} \models \phi$ under LTL semantics, which is easily verified by comparing the respective satisfaction conditions.

We are now ready to introduce the semantics of TEL[Aguado *et al.*, 2023].

Definition 2 (Stable (Equilibrium, TEL) Model)**.** *A trace $\mathbf{T}$ is a stable (equilibrium, TEL) model of formula ϕ if (i) $\mathbf{T} \models \phi$, i.e., $\mathbf{T}$ is an LTL model of ϕ, and (ii) no $\mathbf{H} \neq \mathbf{T}$ exists s.t. $\langle \mathbf{H}, \mathbf{T} \rangle \models \phi$.*

Example 3. *For a predicate $admin(X)$ varying over time and its negation $n\_admin(X)$,* inertia *default rules are as follows:*

$$\Box(admin(X) \wedge \neg\circ n\_admin(X) \rightarrow \circ admin(X)) \qquad (2)$$

$$\Box(n\_admin(X) \wedge \neg\circ admin(X) \rightarrow \circ n\_admin(X)) \qquad (3)$$

(2) *states that, in any situation where* $admin(X)$ *holds and there is no evidence that in the next state* $n\_admin(X)$ *holds, then* $admin(X)$ *remains true;* (3) *is analogous for* $n\_admin(X)$. *The conjunction of* (2)*,* (3)*, and the formula* $\circ\circ admin(X) \wedge \circ\circ\Diamond n\_admin(X)$ *grounded to* $X = u1$ *has the stable models given by the regular expression* $\varnothing \cdot \varnothing \cdot \{admin(u1)\}^+ \cdot \{n\_admin(u1)\} \cdot \{n\_admin(u1)\}^\lambda$*, where* λ *ranges over* $\mathbb{N} \cup \{\omega\}$*, i.e., induces finite and infinite traces.*

3 Eager Unfoldable Operators

Even though TEL effectively captures the intended meaning of scenarios like Example 2, where local positive loops are considered, it struggles to fully capture the intended semantics of Example 1. Consider the following formula:

$$\Box(\Diamond r) \wedge \Box(r \rightarrow \Diamond g). \tag{4}$$

This formula does not admit any infinite stable model. Intuitively, given a candidate LTL model $\mathbf{T}$, it is always possible to construct a smaller $\mathbf{H}$ such that $\langle \mathbf{H}, \mathbf{T}\rangle \vDash \varphi$ by omitting g in H_i for some arbitrary time point i, even if g occurs in T_i. This minimization condition undermines the satisfaction of the formula's intended meaning. The issue persists in the finite case, where the unique stable model involves finitely many occurrences of r, with g appearing only in the last state. Under TEL, any finite LTL model where g appears earlier in the trace is unstable, as we can remove it from $\mathbf{H}$.

A similar argument applies to the sub-formula $\Box(\Diamond r)$ by itself. As r must occur infinitely often, we can always remove an occurrence of r from $\mathbf{H}$ and still have THT-satisfaction. This highlights a fundamental limitation of TEL's strong minimization criterion in scenarios like Example 1.

The operators $\Diamond$ and $\mathbb{U}$ are non-deterministic in Definition 1, as their satisfaction depends on arbitrary points in the trace, akin to temporal disjunction. In contrast, $\Box$ and $\circ$ are deterministic, and evaluated directly at specific points. To address this, we propose eager variants of $\mathbb{U}_e$, $\mathbb{R}_e$, $\mathbb{S}_e$, and $\mathbb{T}_e$, which unfold deterministically once a $\mathbf{T}$-trace is fixed. These operators are equivalent to the standard ones in LTL but weaker in THT, resolving non-determinism for clearer computational and modeling benefits.

Definition 3. *The eager variant* O_e *of the operator* $\mathsf{O} \in \{\mathbb{U}, \mathbb{R}, \mathbb{S}, \mathbb{T}\}$ *is as follows. For any* THT*-trace* $\mathbf{I} = \langle \mathbf{H}, \mathbf{T}\rangle$ *and time point* $k \geq 0$*,*

12. $\mathbf{I}, k \vDash \phi\, \mathbb{U}_e\, \psi$ *if there exists some* $j \geq k$ *s.t.* $\mathbf{I}, j \vDash \psi$*, and for all* $j' \in [k, j)$*,* $\mathbf{I}, j' \vDash \phi$ *and* $\mathbf{T}, j' \nvDash \psi$*;*
13. $\mathbf{I}, k \vDash \phi\, \mathbb{R}_e\, \psi$ *if for all* $j \geq k$ *s.t. (a)* $\mathbf{I}, j \nvDash \psi$ *or (b)* $\mathbf{T}, j \vDash \phi$ *and*$\mathbf{I}, j \nvDash \phi$*, some* $j' \in [k, j)$ *exists s.t.* $\mathbf{I}, j' \vDash \phi$*;*
14. $\mathbf{I}, k \vDash \phi\, \mathbb{S}_e\, \psi$ *if there is some* $j \leq k$ *s.t.* $\mathbf{I}, j \vDash \psi$*, and for all* $j' \in (j, k)$*,* $\mathbf{I}, j' \vDash \phi$ *and* $\mathbf{T}, j' \nvDash \psi$*;*
15. $\mathbf{I}, k \vDash \phi\, \mathbb{T}_e\, \psi$ *if for all* $j \geq k$ *s.t. (a)* $\mathbf{I}, j \nvDash \psi$ *or (b)* $\mathbf{T}, j \vDash \psi$ *and* $\mathbf{I}, j \nvDash \psi$*, some* $j' \in [k, j)$ *exists s.t.* $\mathbf{I}, j' \vDash \phi$*.*

From these new operators, we can derive as usual operators such as: $\Box_e$ (eager globally) by $\Box_e\phi := \bot\, \mathbb{R}_e\, \phi$; $\Diamond_e$ (eager eventually) by $\Diamond_e\phi := \top\, \mathbb{U}_e\, \phi$; $\blacksquare_e$ (eager historically) by $\blacksquare_e\phi := \bot\, \mathbb{T}_e\, \phi$; etc.

The notion of stable model in Definition 2 naturally extends to the new operators. In Example 1, we mentioned that one might be interested in expressing the following properties: (i) every request must eventually be satisfied with a grant, and (ii) a grant is given only if there was a prior request. In other words, intuitively, point (ii) requires a justification for every grant.

With the novel temporal operators, We can effectively capture points (i) and (ii) under stable semantics.

$$\Box(\Diamond_e r) \wedge \Box(r \rightarrow \Diamond_e g). \tag{5}$$

This formula yields infinitely many infinite models, where all occurrences of g along the trace following a request are considered stable. Intuitively, the new operators can be interpreted as suspending the minimization for a stable (Definition 2), introducing a temporal instance of the excluded middle axiom, which is propagated until the temporal formula is fulfilled. Note that if $\langle \mathbf{H}, \mathbf{T}\rangle, i \vDash p \vee \neg p$, then $p \in H_i$ iff $p \in T_i$.

From Corollary 1 we obtain that the new operators are expressible in THT.

Lemma 1. *In* THT*,*

$$\phi\, \mathbb{U}_e\, \psi \equiv (\phi \wedge \neg\psi)\, \mathbb{U}\, \psi \qquad \phi\, \mathbb{R}_e\, \psi \equiv \phi\, \mathbb{R}\, (\psi \wedge (\phi \vee \neg\phi))$$
$$\phi\, \mathbb{S}_e\, \psi \equiv (\phi \wedge \neg\psi)\, \mathbb{S}\, \psi \qquad \phi\, \mathbb{T}_e\, \psi \equiv \phi\, \mathbb{T}\, (\psi \wedge (\phi \vee \neg\phi))$$

Notably, $\Box_e$ coincides with $\Box$ and $\blacksquare_e$ with $\blacksquare$. According to Theorem 1 in [Cabalar and Diéguez, 2014], THT-equivalent formulas ϕ and ψ are *strongly equivalent* in TEL, i.e., replacing any occurrence of ϕ in a formula χ with ψ, or vice versa, results in the same stable models for χ. Consequently, the formula $\tau_e(\phi)$ that results from a formula ϕ by exhaustive application of Lemma 1 has the same stable models as ϕ.

We then obtain the following result.

Theorem 1. *Deciding* TEL*-satisfiabilty of a formula* ϕ *in which eager operators occur is EXPSPACE-complete.*

The membership part follows from EXPSPACE-completeness of TEL-satisfiability and the fact that $\tau_e(\phi)$ and ϕ have the same stable models; an exponential blowup can be avoided using Tseitin-style [Tseitin, 1983] naming of subformulas ψ, i.e., using atoms p_ψ and adding $p_\psi \leftrightarrow \psi$. The EXPSPACE-hardness follows from the fragment with arbitrary nesting of $\Box$ and implication [Bozzelli and Pearce, 2015] as well as from the fragment with $\Box$ and $\circ$ [Šimkus, 2010].

4 Non Disjunctive Temporal Programs

In this section, we introduce $NDTP$ programs, a nondisjunctive, rule-based fragment extending the class considered in [Erdem and Lifschitz, 2003] to the temporal case by incorporating TEL modalities in the body and allowing arbitrary nesting of eager unfoldable operators in the head.

Definition 4 ($NDTP$)**.** *A Non-Disjunctive Temporal Program* π *consists of*

(i) a set $init(\pi)$ *of initial rules of the form*

$$r : \psi \rightarrow \phi \tag{6}$$

where ϕ *is either* $\bot$ *or a* head formula *from the grammar*

$$
\begin{aligned}
\phi &::= \eta[\phi] \mid \phi\ \mathsf{O}_e\ \phi \text{ for } \mathsf{O}_e \in \{\mathbb{U}_e, \mathbb{R}_e, \mathbb{S}_e, \mathbb{T}_e\} \\
\psi &::= \psi_1 \mid \psi_2 \\
\psi_1 &::= \eta[\psi_1] \mid \psi_1 \vee \psi_1 \mid \psi_1\ \mathsf{O}\ \psi_1 \text{ for } \mathsf{O} \in \{\mathbb{U}, \mathbb{R}, \mathbb{S}, \mathbb{T}\} \\
\psi_2 &::= \eta[\psi_2] \mid \neg\gamma \mid \psi_2 \vee \psi_2 \\
\eta[\mu] &::= \top \mid p \mid \circ\eta[\mu] \mid \hat{\circ}\eta[\mu] \mid \bullet\eta[\mu] \mid \hat{\bullet}\eta[\mu] \mid \\
&\quad\ \eta[\mu] \wedge \eta[\mu] \mid \mu
\end{aligned}
$$

where $p \in \mathcal{P}$ and γ is an arbitrary TEL formula, $\eta[\cdot]$ is parameterized by a grammar symbol (ϕ, ψ_1 or ψ_2), and

(ii) a set $dyn(\pi)$ of dynamic rules of the form $\Box r$, where r is an initial rule.

For each rule r we denote ψ as $body(r)$ and ϕ as $head(r)$.

The $NDTP$ fragment disallows: (i) nesting of implications outside the scope of a negation, (ii) negation within temporal unfoldable operators, and (iii) disjunctions in the head. It permits (i) eager operators in the head, and (ii) arbitrary formulas under negation. Notably, all examples from above are in this fragment.

The eager operators unfold deterministically along a given trace due to the restricted temporal disjunctions, making them conceptually similar to "shiftable" operators. This deterministic behavior ensures a clear justification for atoms appearing along a trace $\mathbf{T}$, while the body admits richer syntax under the condition that the negated sub-formulas can be evaluated and replaced with $\top$ or $\bot$ without the need for further unfolding beyond the one for the dynamic rules.

As for properties of $NDTP$, a notable one is that eager operators in rule heads do not eliminate stable models.

Proposition 2 (Equilibrium Persistence)**.** *Given an $NDTP$ program π, let π' be a formula obtained from π by replacing some eager operators O_e with O. Then every stable model of π' is a stable model of π.*

Furthermore, as for ordinary logic programs, negation is necessary to prevent stability. Let us call $\neg$-free $NDTP$ programs *positive*. Then

Theorem 2 (Positive $NDTP$)**.** *Every positive $NDTP$ program π that is LTL-satisfiable has a stable model.*

In particular, if π in Theorem 2 has no heads $\bot$ it has some stable model. For example, recall that $\Box\Diamond_e r$ has infinitely many stable models, while its counterpart $\Box\Diamond r$ has none.

Next, we observe that using a Tseitin-style transformation with auxiliary atoms as mentioned above, it is possible to obtain a normal form for head formulas.

Proposition 3 (Head Normal Form)**.** *Every $NDTP$ program π is rewritable in polynomial time to an $NDTP$ program π' that has the same stable models as π modulo auxiliary atoms where π' has only head formulas of the form p, $\circ p$, and $\bot$.*

This result is particularly useful for simplifying inference and deriving complexity results.

We next present a characterization of stable models in terms of a program reduct.

Definition 5 (λ-Unfolding)**.** *For any $NDTP$ program π and $\lambda \geq 1$, the temporal program π^λ is $\pi^\lambda := init(\pi) \cup \{\circ^i r \mid \Box r \in \pi_{dyn}, i \in [0, \lambda)\}$.*

It is easy to see that π^λ and π share the same LTL, THT and TEL models, for any $NDTP$ program π and $\lambda \geq 1$, and that $\Box r$ holds iff r holds in every state $i \in [0, \lambda)$ according to Defn. 1. We can now introduce temporal program reducts. Denote by $sub_\neg(\phi)$ the maximal subformulas of ϕ of the form $\neg\psi$, i.e., not within the scope of another negation.

Definition 6 (Temporal reduct)**.** *The temporal reduct $\pi^{\mathbf{T}}$ of an $NDTP$ program π w.r.t. an LTL trace $\mathbf{T}$ results from π^λ by (1) replacing every $\circ^i r \in \pi^\lambda$ with $\tau_e(\circ^i r)$; (2) unfolding temporal operators if they are finitely fulfilled; (3) if $\mathbb{R}$ (or $\mathbb{T}$) unfolds infinitely, replace it with $\Box$ (resp. $\blacksquare$); (4) replace each $\psi \in sub_\neg(\circ^i r)$ with $\top$ if $\mathbf{T}, i \vDash \psi$, and with $\bot$ otherwise.*

We note that the temporal reduct is well-defined as negation is not allowed inside unfoldable temporal operators such as $\mathbb{U}, \mathbb{R}, \mathbb{S}, \mathbb{T}$, or their derived operators. The deterministic nature of rule heads in $NDTP$ programs is formally evidenced by the fact that they admit a single minimal Here-Trace. Such a result can be shown by defining a fixed-point operator over a lattice of H-traces using the unfolded reduct $\pi^{\mathbf{T}}$ as rules.

Proposition 4. *Let π be an $NDTP$ program and $\mathbf{T}$ an LTL model of π. Then there exists an $\mathbf{H}$ such that $\mathbf{H} \leq \mathbf{H}'$ whenever $\langle \mathbf{H}', \mathbf{T} \rangle \vDash \pi^{\mathbf{T}}$.*

We call $\mathbf{T}$ a *λ-stable model* of π if $\mathbf{T}$ is an LTL-minimal model of $\pi^{\mathbf{T}}$ (equivalently, $\mathbf{T}$ coincides with $\mathbf{H}$ in Proposition 4). We link the equilibrium-based (Defn. 2) and reduct-based (Defn. 6) stable models, showing that they coincide.

Proposition 5 (Equivalence)**.** *A trace $\mathbf{T}$ is a stable model of an $NDTP$ program π iff $\mathbf{T}$ is a λ-stable model of $\pi^{\mathbf{T}}$.*

Regarding the form of stable models, any TEL-satisfiable formula ϕ that has an infinite stable model has one of the form $\mathbf{T} = \mathbf{T}_P \cdot \mathbf{T}_C^\omega$, where the length of $\mathbf{T}_P \cdot \mathbf{T}_C$ is at most double-exponential in the size of ϕ [Bozzelli and Pearce, 2015]. In the case of $NDTP$ programs, this falls back to a single exponential length, and a finite stable model (if any) of exponential length exists. Details are given in the appendix.

Turning to complexity, it is well known that model-checking of an LTL formula ϕ on a trace $\mathbf{T}$ is feasible in polynomial time [Demri and Schnoebelen, 2002]. This similarly holds for stable model checking of $NDTP$ programs.

Theorem 3. *Given an $NDTP$ program π and a (finitely represented) trace $\mathbf{T}$, deciding whether $\mathbf{T}$ is a stable model of π is feasible in polynomial time.*

As for deciding TEL-satisfiability of $NDTP$ programs, we have the following result.

Theorem 4. *Deciding whether an $NDTP$ program π is TEL-satisfiable is in EXPTIME.*

Proof. (Sketch) The proof proceeds by encoding the program π into a Büchi automata using well-established LTL-to-automata techniques. To address the two-world semantics of THT, we employ the *star-translation*, which maps the two-world structure into single-world representation by introducing fresh atoms for the Here-world formulas.

The automata construction involves two components: the (i) *HT-local automata*, which enforces local consistency for the There-world, and closure for head and body formulas for

the Here-world. So if p, q are in the Here part of a node, and $p \wedge q$ appears as a sub-formula in π, then we want also $p \wedge w$ in the Here part of a node; the (ii) *T-eventuality automata*, which imposes fairness conditions, requiring that if an $\mathbb{U}$ formula is derived infinitely often, it must also be fulfilled infinitely often. We compute the synchronous product of these two automata.

The minimality of the $\mathbf{H}$-trace is achieved via a fix-point computation. Starting with a list of nodes where nothing positive is derived in the Here-part, we iteratively apply the immediate consequence operator, propagating head formulas through the list of nodes once derived. Once a set of head formulas is derived, we update the list of nodes with new nodes with the minimal set of formulas in the Here part such that they contain the derived formulas. Identifying such nodes is deterministic due to the syntactic restrictions of $NDTP$ programs. □

Consequently, $NDTP$ programs have, besides appealing semantic properties, significantly lower complexity than the full language of TEL. A matching lower bound is open, while PSPACE-membership of the problem is plausible.

5 Tight Temporal Programs

In this section, we introduce tight temporal programs (TTP), a fragment of TEL designed to keep complexity low by relying on LTL solvers. In ASP, the notion of tightness – programs without positive recursive dependencies – ensures that stable models can be obtained as supported models via Clark's completion. We establish similar results for $NDTP$ programs, despite the challenges of defining support when (nested) temporal operators are allowed.

We start by defining the concept of support for head formulas, which will serve as the foundation for introducing support in $NDTP$ programs.

Definition 7 (Supporting Head)**.** *Given a head formula φ, a trace $\mathbf{T}$, and $i, j \in [0, \lambda_{\mathbf{T}})$, we say that $p \in T_i$ is supported by φ at time j, denoted $supp_{\mathbf{T}}(\varphi, j, p, i)$, if $\mathbf{T}, j \vDash \varphi$ and*

- *$\varphi = p$: $i = j$ and $i < \lambda_{\mathbf{T}}$;*
- *$\varphi = \varphi_1 \wedge \varphi_2$: $supp(\varphi_k, j, p, i)$ for some $k \in \{1, 2\}$;*
- *$\varphi = \varphi_1 \mathbb{U}_e \varphi_2$: if $\mathbf{T}, j \vDash \varphi_2$ then $supp_{\mathbf{T}}(\varphi_2, j, p, i)$ else either $supp_{\mathbf{T}}(\varphi_1, j, p, i)$ or $supp_{\mathbf{T}}(\varphi, j+1, p, i)$;*
- *$\varphi = \varphi_1 \mathbb{R}_e \varphi_2$: if $\mathbf{T}, j \vDash \varphi_1$ then either $supp_{\mathbf{T}}(\varphi_1, j, p, i)$ or $supp_{\mathbf{T}}(\varphi_2, j, p, i)$, else either $supp_{\mathbf{T}}(\varphi_2, j, p, i)$ or $supp_{\mathbf{T}}(\varphi, j+1, p, i)$;*
- *$\varphi = \varphi_1 \mathbb{S}_e \varphi_2$: if $\mathbf{T}, j \vDash \varphi_2$ then $supp_{\mathbf{T}}(\varphi_2, j, p, i)$ else either $supp_{\mathbf{T}}(\varphi_1, j, p, i)$ or (b2) $supp_{\mathbf{T}}(\varphi, j+1, p, i)$;*
- *$\varphi = \varphi_1 \mathbb{T}_e \varphi_2$: if $\mathbf{T}, j \vDash \varphi_1$ then either $supp_{\mathbf{T}}(\varphi_1, j, p, i)$ or $supp_{\mathbf{T}}(\varphi_2, j, p, i)$ else either $supp_{\mathbf{T}}(\varphi_2, j, p, i)$ or $supp_{\mathbf{T}}(\varphi, j-1, p, i)$;*
- *$\varphi = \circ\varphi_1$ or $\varphi = \hat{\circ}\varphi_1$: $supp_{\mathbf{T}}(\varphi_1, j+1, p, i)$;*
- *$\varphi = \bullet\varphi_1$ or $\varphi = \hat{\bullet}\varphi_1$: $supp_{\mathbf{T}}(\varphi_1, j-1, p, i)$.*

Furthermore, $supp_{\mathbf{T}}(\phi, j) = \{(p, i) \mid supp_{\mathbf{T}}(\phi, j, p, i)\}$.

Note that Definition 7 is more straightforward in the atemporal case unless one wants to cover also boolean operators in the head, see e.g. [Alviano *et al.*, 2016].

Definition 8 (Supported LTL Models)**.** *Trace $\mathbf{T}$ is supported by an $NDTP$ π, if for every $j \in [0, \lambda_{\mathbf{T}})$ and $p \in T_j$, either some (a) $\Box r \in dyn(\pi)$ or (b) $r \in init(\pi)$ exists s.t $\mathbf{T}, i \vDash body(r)$ and $supp_{\mathbf{T}}(head(r), i, p, j)$, where $i = 0$ in case (b).*

We notice that the stable models of the theory in Example 3 coincide with its supported models. More in general, as for the relationship to LTL- and TEL semantics, we show that supported LTL models relax TEL-models.

Proposition 6. *Every stable model $\mathbf{T}$ of an $NDTP$ program π is a supported LTL model of π.*

Furthermore, the supported LTL-models can be expressed in LTL. To this end, we introduce a temporal version tcc of Clark's completion [Clark, 1977] such that $\mathbf{T}$ is a supported model of π iff $\mathbf{T}$ is an LTL model of $tcc(\pi)$.

Definition 9 (Temporal Clark's Completion)**.** *For any $NDTP$ π, its temporal completion is*

$$tcc(\pi) = \pi \wedge \bigwedge_{p \in \mathcal{P} \cup \{\top\}} \Box\gamma(p), \tag{7}$$

where $\gamma(p) := p \rightarrow \bigvee_{r \in \pi} justify_r^p(head(r), body(r))$ and $justify_r^p(\phi, \psi)$ is inductively defined as follows:

- *$\phi = p$: if $r \in init(\pi)$ then $\neg\bullet\top \wedge \psi$ else ψ;*
- *$\phi = q \neq p$, $\phi = \top$, or $\phi = \bot$: $\bot$;*
- *$\phi = \phi_1 \wedge \phi_2$: $justify_r^p(\phi_1, \psi) \vee justify_r^p(\phi_2, \psi)$;*
- *$\phi = \phi_1 \,\mathsf{O}_e\, \phi_2$, for $\mathsf{O} \in \{\mathbb{U}, \mathbb{R}, \mathbb{S}, \mathbb{T}\}$: $\gamma_{\mathsf{O},1} \vee \gamma_{\mathsf{O},2}$, where*

$$\begin{aligned}
\gamma_{\mathbb{U},1} &= justify_r^p(\phi_2, \phi_2 \wedge (\psi \vee \bullet((\psi \wedge \neg\phi_2)\ \mathbb{S}\ \psi)),\\
\gamma_{\mathbb{U},2} &= justify_r^p(\phi_1, \neg\phi_2\ \mathbb{S}\ (\psi \wedge \neg\phi_2));\\
\gamma_{\mathbb{R},1} &= justify_r^p(\phi_1, \phi_1 \wedge (\psi \vee \bullet(\neg\phi_1\ \mathbb{S}\ (\neg\phi_1 \wedge \psi)))),\\
\gamma_{\mathbb{R},2} &= justify_r^p(\phi_2, \psi \vee \bullet(\neg\phi_1\ \mathbb{S}\ (\neg\phi_1 \wedge \psi)));\\
\gamma_{\mathbb{S},1} &= justify_r^p(\phi_2, \phi_2 \wedge (\psi \vee \circ((\psi \wedge \neg\phi_2)\ \mathbb{U}\ \psi))\\
\gamma_{\mathbb{S},2} &= justify_r^p(\phi_1, \neg\phi_2\ \mathbb{U}\ (\psi \wedge \neg\phi_2));\\
\gamma_{\mathbb{T},1} &= justify_r^p(\phi_1, \phi_1 \wedge (\psi \vee \circ(\neg\phi_1\ \mathbb{U}\ (\neg\phi_1 \wedge \psi))))\\
\gamma_{\mathbb{T},2} &= justify_r^p(\phi_2, \psi \vee \circ(\neg\phi_1\ \mathbb{U}\ (\neg\phi_1 \wedge \psi)));
\end{aligned}$$

- *$\phi = \circ\phi_1$, $\phi = \hat{\circ}\phi_1$: $\bullet justify_r^p(\phi_1, \psi)$,*
- *$\phi = \bullet\phi_1$, $\phi = \hat{\bullet}\phi_1$: $\circ justify_r^p(\phi_1, \psi)$.*

Temporal operators like $\mathbb{U}_e$, $\mathbb{S}_e$, $\circ$ introduce dependencies between states, meaning that the support for an atom might need to be propagated among states.

Example 4. Consider the $NDTP$ π with two dynamic rules $\Box(r_1)$ and $\Box(r_2)$, where $r_1 = \circ p \rightarrow p$ and $r_2 = \Box(\circ\top)$, and no initial rules. As easily seen, $\Box(r_2)$ enforces infinite models; we are thus in the infinitary (standard) setting of LTL. Applying the temporal Clark's completion, we obtain $\gamma(p) = p \rightarrow \circ p \vee \bot$. Thus $tcc(\pi) = \pi \cup \{\Box(p \rightarrow \circ p \vee \bot)\}$, which has the LTL models $\mathbf{T} = \{p\}^\omega$ and $\mathbf{H} = \varnothing^\omega$.

In Example 4, $\mathbf{T}$ is supported by π but not a TEL-model of π (as $\langle \mathbf{H}, \mathbf{T} \rangle \vDash \pi$). However, without r_2, finite models of $tcc(\pi)$ would coincide with stable models. In general, we have:

Proposition 7 (Characterization)**.** *For any $NDTP$ program π and an LTL model of π, $\mathbf{T} \vDash tcc(\pi)$ iff $\mathbf{T}$ is a supported LTL model of π.*

On the other hand, the LTL semantics of a program π is easily recovered from the supported LTL models of an augmented program, by adding $\Box(p \rightarrow p)$ for each $p \in \mathcal{P}$ to π.

We remark that the completion formula $tcc(\pi)$ can be exponential in the nesting depth of operators O_e in rule heads. For a polynomial LTL encoding, we can introduce Tseitin-style naming of subformulas as described above.

We next introduce notions of parents and relevance.

Definition 10. *For the unfolded program $\pi^{\mathbf{T}}$ of an NDTP program π and a trace $\mathbf{T}$, an atom p^i is a* parent *of q^j if some rule $\circ^k r$ exists s.t. $\mathbf{T}, i \vDash body(r)$, $supp_{\mathbf{T}}(head(r), k, q, j)$ and p^i is relevant for $body(r)$ w.r.t. $\mathbf{T}$. We call p^i* relevant *for $body(r)$ w.r.t. $\mathbf{T}$ when in checking recursively $\mathbf{T}, k \vDash body(r)$ (w.r.t. Defn. 1) we may end up evaluating $\mathbf{T}, i \vDash p$.*

Note that given a head formula ϕ, $(p, i) \in supp_{\mathbf{T}}(\phi, j)$ iff p^i is relevant for ϕ w.r.t. $\mathbf{T}$.

In the following definition, we use the "being parent" relation over atoms. We assume that there is an order over the atoms in $\mathcal{P}$, and we use the subscript k in p^i_k to refer to the k-th atom in $\mathcal{P}$ at time point i.

Definition 11. *An NDTP program π is tight on $\mathbf{T}$, if there are no infinite sequences $p^{i_0}_{j_0}, p^{i_1}_{j_1}, \ldots$ such that for each $k \geq 1$, $p^{i_k}_{j_k}$ is a parent of $p^{i_{k-1}}_{j_{k-1}}$.*

We recall and accommodate Proposition 3 from [Erdem and Lifschitz, 2003] to our terminology:

Proposition 8. *A program π is tight on a trace $\mathbf{T}$ iff there exists a function f from $\mathcal{P} \times [0, \lambda_{\mathbf{T}})$ to ordinals such that for every rule $\circ^i$ in $\pi^{\mathbf{T}}$ with $\mathbf{T}, i \vDash body(r) \wedge head(r)$, for every $(p, i) \in supp_{\mathbf{T}}(head(r), j)$, and for every q^k relevant for $body(r)$ w.r.t $\mathbf{T}$, we have $f(p, j) < f(q, k)$.*

Based on this, we can then show:

Theorem 5. *For any NDTP program π and any LTL model of π tight on $\mathbf{T}$, $\mathbf{T}$ is an stable trace for π iff $\mathbf{T}$ is supported by π iff $\mathbf{T} \vDash \pi \wedge tcc(\pi)$.*

We proceed to a variant of Theorem 5 that is not parameterized by an LTL model $\mathbf{T}$, but works for all possible traces. To achieve this goal, we need a notion of dependence among atoms that does not rely on a specific trace.

The dependency graph is an important tool in logic programming and is the basis of many relevant results, among them the semantic properties of tight logic programs. For our concerns, the dependency graph is defined as follows.

Definition 12 (Dependency Graph)**.** *Given an π be a NDTP program, we define its dependency graph, denoted $DG(\pi) = \langle V, E \rangle$ where:*

- *Each vertex $v \in V$ corresponds to an indexed subformula ϕ^j in the ω-unfolding π^ω of π, representing the instance of ϕ holding at position j in the trace.*
- *The arcs in E are defined recursively as follows:*
  - *head-to-body links:*
  - *for $\circ^i r$ in π^ω, E has an arc $head(r)^i \rightarrow body(r)^i$;*
  - *head formulas dependencies:*
  - *an arc from $(\phi_1 \wedge \phi_2)^i$ to ψ^j implies an arc $\phi^i_k \rightarrow \psi^j$;*
  - *an arc $\phi^i \rightarrow \psi^j$ resp. $\hat{\circ}\phi^i \rightarrow \psi^j$ implies an arc $\phi^{i+1} \rightarrow \psi^j$;*
  - *an arc $(\bullet\phi)^i \rightarrow \psi^j$ resp. $(\hat{\bullet}\phi)^i \rightarrow \psi^j$ implies an arc $\phi^{i-1} \rightarrow \psi^j$ if $i - 1 \geq 0$;*
  - *an arc $(\phi_1 \, \mathsf{O}_e \, \phi_2)^i \rightarrow \psi^j$, for $\mathsf{O}_e \in \{\mathbb{U}_e, \mathbb{R}_e, \mathbb{S}_e, \mathbb{T}_e\}$), implies an arc $\phi^s_k \rightarrow \psi^j$ for every $k \in \{1, 2\}$ and every $s \geq i$ if $\mathsf{O}_e \in \{\mathbb{U}_e, \mathbb{R}_e\}$ and $s \in [0, j]$ otherwise;*
  - *body formulas dependencies:*
  - *an arc $\phi^i \rightarrow (\psi_1 \vartriangle \psi_2)^j$, for $\vartriangle \in \{\vee, \wedge\}$, implies an arc $\phi^i \rightarrow \psi^j_k$ for $k \in \{1, 2\}$;*
  - *an arc $\phi^i \rightarrow \circ(\psi)^j$ resp. $\phi^i \rightarrow \hat{\circ}(\psi)^j$ implies an arc $\phi^i \rightarrow \psi^{j+1}$;*
  - *an arc $\phi^i \rightarrow \bullet(\psi)^j$ resp. $\phi^i \rightarrow \hat{\bullet}(\psi)^j$ implies an arc $\phi^i \rightarrow \psi^{j-1}$ if $j - 1 \geq 0$;*
  - *an arc $\phi^i \rightarrow (\psi_1 \, \mathsf{O} \, \psi_2)^j$ for $\mathsf{O} \in \{\mathbb{U}, \mathbb{R}, \mathbb{S}, \mathbb{T}\}$) implies an arc $\phi^i \rightarrow \psi^s_k$ for every $k \in \{1, 2\}$ and every $s \geq i$ if $\mathsf{O} \in \{\mathbb{U}, \mathbb{R}\}$ and $s \in [0, j]$; otherwise;*

Intuitively, the construction parses each rule in π to extract its head and body, analyzing recursively its structure. As customary in logic programming, the transitive closure of $DG(\pi)$ represents the positive dependencies in program π.

Definition 13 (Tight Temporal Program)**.** *A* tight temporal program *(TTP) is an $NDTP$ program π that is* tight*, which means that $DG(\pi)$ has no infinite path.*

Definition 13 ensures the following property.

Lemma 2. *If an $NDTP$ program π is tight, then it is tight w.r.t. any possible trace $\mathbf{T}$.*

As a corollary of Theorem 5 and Lemma 2, we then obtain:

Theorem 6. *Let π be a TTP program π. Then $\mathbf{T}$ is a TEL-model of π iff $\mathbf{T}$ is a supported LTL-model of π iff $\mathbf{T} \vDash \pi \wedge tcc(\pi)$.*

From Theorem 6 and the fact that the Clark completion $tcc(\pi)$ is constructible in polynomial time, we thus obtain the following complexity result.

Theorem 7 (Satisfiability)**.** *Deciding whether a given TTP program π has a stable model is PSPACE-complete.*

The membership comes from LTL, which is well-known to be PSPACE-complete. The PSPACE-hardness holds as we can express an LTL formula ϕ by $\neg\phi \rightarrow \bot$ and imposing $\Box(\neg\neg p \rightarrow p)$ for each atom p; note that $\neg\neg p \rightarrow p$ is in THT equivalent to $\neg p \vee p$ (excluded middle).

Thus, for this class of programs TEL-satisfiability has the same complexity as classical LTL-satisfiability. However, to exploit this, we must check for tightness, which could be expensive. Fortunately, it turns out that this is not the case.

Deciding Tightness. We utilize a finite representation of the temporal dependency graph $DG(\pi)$ in terms of a *one-counter automaton* [Comon and Jurski, 1998]. Such an automaton is a non-deterministic finite-state automaton operating on a single counter variable with value increments from $\{-1, 0, 1\}$.

Definition 14 (One-Counter Automaton)**.** *A one-counter automaton (OCA) A is a tuple (Q, Q_{t_0}, Δ), where:*

- *Q is a finite set of control states,*

- $Q_{t_0} \subseteq Q$ *is a subset of control states,*
- $\Delta \subseteq Q \times Q \times \{-1, 0, 1\}$ *is a finite transition relation.*

Informally, Q_{t_0} represents the set of states that can only be accessed if the counter equals zero. A *configuration* of an OCA is a pair (q, c) of a state $q \in Q$ and an integer $c \geq 0$ that represents the current value of the counter. The transition relation $(q_1, c_1) \to (q_2, c_2)$ between configurations holds if some $(q_1, q_2, c) \in \Delta$ exists such that $c_2 = c_1 + c$, where in case $q_i \in Q_{t_0}$ we must have $c_i = 0$, for $i \in \{1, 2\}$.

A *path* from a configuration (q_1, c_1) to a configuration (q_2, c_2), denoted by $\pi : (q_1, c_1) \to^* (q_2, c_2)$, is a sequence of transitions leading from (q_1, c_1) to (q_2, c_2). We represent the dependency graph of a program π by an OCA $A_{DG}(\pi) = \langle Q, Q_0, \Delta \rangle$ as follows:

- $Q = Q^b \cup Q^h \cup \mathcal{P}$, where Q^b (resp. Q^h) contains ϕ^b (ϕ^h) for each sub-formula ϕ in the body (head) of some rule r in π, where we let $p^b = p^b = p$ for each atom $p \in \mathcal{P}$;
- $Q_0 = \{body(r)^b, head(r)^h \mid r \in init(\pi)\}$;
- Δ is defined recursively akin to E for $DG(\pi)$, intuitively with a level increase or decrease in the ω-unfolding of the program π. It contains
  - head-to-body links:
  - $(head(r)^h, body(r)^b, 0)$, for each rule $r \in init(\pi) \cup dyn(\pi)$;
  - head formulas dependencies: for each $\gamma^h \in Q$ of form
  - $(\phi_1^h \wedge \phi_2^h)^h$: $(\phi_k^k, \gamma^h, 0)$;
  - $(\circ\phi^h)^h, (\hat{\circ}\phi^h)^h$: $(\phi^h, \gamma^h, -1)$;
  - $(\bullet\phi^h)^h, (\hat{\bullet}\phi^h)^h$: $(\phi^h, \gamma^h, +1)$;
  - $(\phi_1^h \, \mathsf{O}_e \, \phi_2^h)^h$, $\mathsf{O}_e \in \{\mathbb{U}_e, \mathbb{R}_e\}$): $(\gamma^h, \gamma^h, +1)$, $(\gamma^h, \phi_k^h, 0)$ for $k \in \{1, 2\}$;
  - $(\phi_1^h \, \mathsf{O}_e \, \phi_2^h)$, $\mathsf{O}_e \in \{\mathbb{S}_e, \mathbb{T}_e\}$): $(\gamma^h, \gamma^h, -1)$, $(\gamma^h, \phi_k^h, 0)$ for $k \in \{1, 2\}$;
  - body formulas dependencies: for each $\gamma^b \in Q$ of form
  - $(\phi_1^b \vartriangle \phi_2^b)^b$, $\vartriangle \in \{\wedge, \vee\}$: $(\gamma^b, \phi_k^b, 0)$ for $k \in \{1, 2\}$;
  - $(\circ\phi^b)^b, \hat{\circ}\phi^b) \in Q$: $(\gamma^b, \phi^b, +1)$;
  - $(\bullet\phi^b)^b, (\hat{\bullet}\phi^b)^b \in Q$: $(\gamma^b, \phi^b, -1)$;
  - $(\phi_1^b \, \mathsf{O} \, \phi_2^b)^b$, $\mathsf{O} \in \{\mathbb{U}, \mathbb{R}\}$: $(\gamma^b, \gamma^b, -1)$, $(\gamma^b, \phi_k^b, 0) \in \Delta$ for $k \in \{1, 2\}$;
  - $(\phi_1^b \, \mathsf{O} \, \phi_2^b)^b$, $\mathsf{O} \in \{\mathbb{S}, \mathbb{T}\}$: $(\gamma^b, \gamma^b, +1)$, $(\gamma^b, \phi_k^b, 0)$ for $k \in \{1, 2\}$.

Intuitively, $A_{DG}(\pi)$ keeps track of the dependencies between atoms in the dependency graph, by emulating arcs with counter increments resp. decrements. It faithfully represents $DG(\pi)$ as follows.

Proposition 9 (OCA Correspondence)**.** *For any NDTP program π and atoms p^i, q^j, it holds that p^i reaches q^j in $DG(\pi)$ iff $(p, i) \to^* (q, j)$ in $A_{DG}(\pi)$.*

We now exploit Proposition 9 to conclude that

Theorem 8. *Deciding whether a given NDTP program π is tight for either finite or infinite semantics is in NL.*

Proof. (Sketch) By Proposition 9, we can use the automaton $A_{DG}(\pi)$ instead of $DG(\pi)$ and distinguishes cases.

Finitary case. We can only have infinite paths by looping. To check this, we can search for a loop under the following two restrictions: (i) we start from a non-deterministically chosen $p \in \mathcal{P}$ with a counter value 0, i.e, from $(p, 0)$, and (ii) we limit the possible configurations to those with a counter value at most $n^3 + n^2$, where n is the number of nodes in $A_{DG}(\pi)$. The bound can be obtained by adjusting a result from [Lafourcade *et al.*, 2005] to our needs.

Infinitary case. Besides loop checking, we need to check for infinite paths with no configuration repetition. We first notice that there is such a path iff there exists a path of length n with an atom p such that there is a path from (p, i) to (p, j) with $i < j$. Therefore, one can try to search for a path where (i) the path passes by an initial configuration at least once and (ii) a path where no initial configurations are visited.

The NL membership follows then as we can construct $A_{DG}(\pi)$ using logarithmic workspace; used as an oracle, we can check in NL whether $A_{DG}(\pi)$ has a cycle. □

6 Conclusion

Recent efforts aim to bridge ASP and Linear Temporal Logic (LTL), in particular in the context of planning [Cabalar *et al.*, 2019] and monitoring [Soldà *et al.*, 2023]. Motivated by issues of the strong minimization criterion for stable models, we introduced eager unfolding temporal operators and defined non-disjunctive temporal programs ($NDTP$) and tight temporal programs (TTP), which amount to fragments of TEL. The latter can be readily translated in polynomial time into LTL using a temporal Clark completion, achieving an exponential complexity drop compared to full TEL and opening a possibility to use LTL-based model checkers.

Temporal Answer Set Programs (TASP) [Aguado *et al.*, 2023] are a rule-based fragment of TEL that has been conceived as an extension of ASP. However, despite a rather plain form of rules, such programs harbour the full complexity of TEL [Cabalar, 2010] which is due to disjunction and $\Diamond$ in rule heads. This similarly follows from results about bidirectional answer set programs [Eiter and Šimkus, 2009; Šimkus, 2010] with a single function symbol, which amount to a fragment of TASP involving only $\circ$ and $\Box$ as temporal operators; excluding disjunction in rule heads results in PSPACE-complexity. Our $NDTP$ programs cater for a much richer syntax, with tight programs allowing for a polynomial translation into LTL.

Walega *et al.* [2021] also considered temporal dependency graphs, yet more for imposing a syntactic constraint to ensure that an immediate consequence operator reaches a fix-point in finite time; forward propagation, for instance, is ruled out. Beck *et al.* [2018] defined support of atoms from rules with temporal operators and complex formulas in the head. However, their notion is semantic without structural decomposition of formulas, requesting that turning the supported atom to false will violate the rule.

Future work will be devoted to implementation and application of the results for monitoring, as well studying possible restrictions and extensions of the program classes introduced.

Ethical Statement

There are no ethical issues.

Acknowledgments

The project leading to this application has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 101034440.

Furthermore, this research was funded in whole or in part by the Vienna Science and Technology Fund (WWTF) project ICT22-023 and the Austrian Science Fund (FWF) project 10.55776/COE12.

References

[Aguado *et al.*, 2023] Felicidad Aguado, Pedro Cabalar, Martín Diéguez, Gilberto Pérez, Torsten Schaub, Anna Schuhmann, and Concepción Vidal. Linear-time temporal answer set programming. *Theory and Practice of Logic Programming*, 23(1):2–56, 2023.

[Alviano *et al.*, 2016] Mario Alviano, Carmine Dodaro, et al. Completion of disjunctive logic programs. In *IJCAI*, volume 16, pages 886–892, 2016.

[Beck *et al.*, 2018] Harald Beck, Minh Dao-Tran, and Thomas Eiter. Lars: A logic-based framework for analytic reasoning over streams. *Artificial Intelligence*, 261:16–70, 2018.

[Bozzelli and Pearce, 2015] Laura Bozzelli and David Pearce. On the complexity of temporal equilibrium logic. In *2015 30th Annual ACM/IEEE Symposium on Logic in Computer Science*, pages 645–656. IEEE, 2015.

[Brewka *et al.*, 2011] Gerhard Brewka, Thomas Eiter, and Miroslaw Truszczynski. Answer set programming at a glance. *Commun. ACM*, 54(12):92–103, 2011.

[Cabalar and Diéguez, 2014] Pedro Cabalar and Martín Diéguez. Strong equivalence of non-monotonic temporal theories. In *Fourteenth International Conference on the Principles of Knowledge Representation and Reasoning*, 2014.

[Cabalar *et al.*, 2019] Pedro Cabalar, Roland Kaminski, Philip Morkisch, and Torsten Schaub. telingo= asp+ time. In *International Conference on Logic Programming and Nonmonotonic Reasoning*, pages 256–269. Springer, 2019.

[Cabalar, 2010] Pedro Cabalar. A normal form for linear temporal equilibrium logic. In *European Workshop on Logics in Artificial Intelligence*, pages 64–76. Springer, 2010.

[Cavada *et al.*, 2014] Roberto Cavada, Alessandro Cimatti, Michele Dorigatti, Alberto Griggio, Alessandro Mariotti, Andrea Micheli, Sergio Mover, Marco Roveri, and Stefano Tonetta. The nuxmv symbolic model checker. In *Computer Aided Verification: 26th International Conference, CAV 2014, Held as Part of the Vienna Summer of Logic, VSL 2014, Vienna, Austria, July 18-22, 2014. Proceedings 26*, pages 334–342. Springer, 2014.

[Clark, 1977] Keith L Clark. Negation as failure. In *Logic and data bases*, pages 293–322. Springer, 1977.

[Comon and Jurski, 1998] Hubert Comon and Yan Jurski. Multiple counters automata, safety analysis and presburger arithmetic. In *Computer Aided Verification: 10th International Conference, CAV'98 Vancouver, BC, Canada, June 28–July 2, 1998 Proceedings 10*, pages 268–279. Springer, 1998.

[Demri and Schnoebelen, 2002] Stéphane Demri and Philippe Schnoebelen. The complexity of propositional linear temporal logics in simple cases. *Information and Computation*, 174(1):84–103, 2002.

[Dodaro *et al.*, 2023] Carmine Dodaro, Giuseppe Mazzotta, and Francesco Ricca. Compilation of tight asp programs. In *ECAI 2023*, pages 557–564. IOS Press, 2023.

[Eiter and Šimkus, 2009] Thomas Eiter and Mantas Šimkus. Bidirectional answer set programs with function symbols. In C. Boutilier, editor, *Proceedings of the 21st International Joint Conference on Artificial Intelligence (IJCAI-09)*, pages 765–771. AAAI Press/IJCAI, 2009.

[Erdem and Lifschitz, 2003] Esra Erdem and Vladimir Lifschitz. Tight logic programs. *Theory and Practice of Logic Programming*, 3(4-5):499–518, 2003.

[Falkner *et al.*, 2018] Andreas Falkner, Gerhard Friedrich, Konstantin Schekotihin, Richard Taupe, and Erich C Teppan. Industrial applications of answer set programming. *KI-Künstliche Intelligenz*, 32(2):165–176, 2018.

[Geatti *et al.*, 2021] Luca Geatti, Nicola Gigante, Angelo Montanari, et al. Black: A fast, flexible and reliable ltl satisfiability checker. In *CEUR Workshop Proceedings*, volume 2987, pages 7–12. CEUR-WS, 2021.

[Holzmann, 1997] Gerard J. Holzmann. The model checker spin. *IEEE Transactions on software engineering*, 23(5):279–295, 1997.

[Lafourcade *et al.*, 2005] Pascal Lafourcade, Denis Lugiez, and Ralf Treinen. Intruder deduction for ac-like equational theories with homomorphisms. In *Term Rewriting and Applications: 16th International Conference, RTA 2005, Nara, Japan, April 19-21, 2005. Proceedings 16*, pages 308–322. Springer, 2005.

[Lifschitz, 2019] Vladimir Lifschitz. *Answer set programming*, volume 3. Springer Heidelberg, 2019.

[Lin and Zhao, 2003] Fangzhen Lin and Jicheng Zhao. On tight logic programs and yet another translation from normal logic programs to propositional logic. In *International Joint Conference on Artificial Intelligence*, 2003.

[McCarthy and Hayes, 1981] John McCarthy and Patrick J Hayes. Some philosophical problems from the standpoint of artificial intelligence. In *Readings in artificial intelligence*, pages 431–450. Elsevier, 1981.

[Pnueli, 1977] Amir Pnueli. The temporal logic of programs. In *18th annual symposium on foundations of computer science (sfcs 1977)*, pages 46–57. ieee, 1977.

[Šimkus, 2010] Mantas Šimkus. *Nonmonotonic logic programs with function symbols*. PhD thesis, Technische Universität Wien, 2010.

[Soldà *et al.*, 2023] Davide Soldà, Ignacio D Lopez-Miguel, Ezio Bartocci, and Thomas Eiter. Progression for monitoring in temporal asp. In *ECAI 2023*, pages 2170–2177. IOS Press, 2023.

[Tseitin, 1983] Grigori S. Tseitin. On the complexity of derivation in propositional calculus. *Automation of reasoning: 2: Classical papers on computational logic 1967–1970*, pages 466–483, 1983.

[Walega *et al.*, 2021] Przemyslaw A. Walega, Michal Zawidzki, and Bernardo C. Grau. Finitely materialisable datalog programs with metric temporal operators. In Meghyn Bienvenu, Gerhard Lakemeyer, and Esra Erdem, editors, *Proceedings of the 18th International Conference on Principles of Knowledge Representation and Reasoning, KR 2021, Online event, November 3-12, 2021*, pages 619–628, 2021.