Loop Formulas for Splitable Temporal Logic Programs*

Felicidad Aguado, Pedro Cabalar,
Gilberto Pérez and Concepción Vidal

Department of Computing,
University of Corunna (Spain)
{aguado,cabalar,gperez,eicovima}@udc.es

Abstract. In this paper, we study a method for computing *temporal equilibrium models*, a generalisation of stable models for logic programs with temporal operators, as in Linear Temporal Logic (LTL). To this aim, we focus on a syntactic subclass of these temporal logic programs called *splitable* and whose main property is satisfying a kind of "future projected" dependence present in most dynamic scenarios in Answer Set Programming (ASP). Informally speaking, this property can be expressed as "past does not depend on the future." We show that for this syntactic class, temporal equilibrium models can be captured by an LTL formula, that results from the combination of two well-known techniques in ASP: splitting and loop formulas. As a result, an LTL model checker can be used to obtain the temporal equilibrium models of the program.

1 Introduction

Although transition systems frequently appear in scenarios and applications of Non-Monotonic Reasoning (NMR), most NMR formalisms are not particularly thought for temporal reasoning. Instead, NMR approaches are typically "static," in the sense that time instants are treated as one more argument for predicates representing actions and fluents. This has been usual, for instance, when representing temporal scenarios in Answer Set Programming (ASP) [1, 2], a successful NMR paradigm based on the *stable model* semantics [3] for logic programs. In this case, it is frequent that program rules depend on a parameter T, the previous situation, and the value $T+1$ representing its successor state. For instance, if t represents the action of toggling a switch that can take two positions, d (down) and u (up), the corresponding *effect axioms* would be encoded as:

$$u(T+1) \leftarrow t(T), d(T) \tag{1}$$

$$d(T+1) \leftarrow t(T), u(T) \tag{2}$$

Similarly, the *inertia law* would typically look like the pair of rules:

$$u(T+1) \leftarrow u(T), \mathit{not}\ d(T+1) \tag{3}$$

$$d(T+1) \leftarrow d(T), \mathit{not}\ u(T+1) \tag{4}$$

* This research was partially supported by Spanish MEC project TIN2009-14562-C05-04 and Xunta de Galicia project INCITE08-PXIB105159PR.

Since ASP tools are constrained to finite domains, a finite bound n for the number of transitions is usually fixed, so that the above rules are grounded for $T = 0, \dots, n-1$. To solve a planning problem, for instance, we would iterate multiple calls to some ASP solver and go increasing the value of $n = 1, 2, 3, \dots$ in each call, until a (minimal length) plan is found.

Of course, this strategy falls short for many temporal reasoning problems, like proving the non-existence of a plan, or checking whether two NMR system representations are *strongly equivalent*, that is, whether they always have the same behaviour, even after adding some common piece of knowledge, and *for any narrative length* we consider.

To overcome these limitations, [4] introduced an extension of ASP to deal with modal temporal operators. Such an extension was the result of mixing two logical formalisms: (1) *Equilibrium Logic* [5, 6] that widens the concept of stable models to the general syntax of arbitrary theories (propositional and even first order); and (2) the well-known *Linear Temporal Logic* [7] (LTL) dealing with operators like $\Box$ (read as "always"), $\Diamond$ ("eventually"), $\bigcirc$ ("next"), $\mathcal{U}$ ("until") and $\mathcal{R}$ ("release"). The result of this combination received the name of *Temporal Equilibrium Logic* (TEL). As happens with Equilibrium Logic, TEL is defined in terms of a monotonic formalism, in this case called *Temporal Here-and-There* (THT), plus an ordering relation among models, so that only the minimal ones are selected (inducing a non-monotonic consequence relation). These minimal models receive the name of *Temporal Equilibrium Models* and can be seen as the analogous of *stable models* for the temporal case. As an example, the rules (1)-(4) can be respectively encoded in TEL as:

$$\Box(d \wedge t \rightarrow \bigcirc u) \tag{5}$$
$$\Box(u \wedge t \rightarrow \bigcirc d) \tag{6}$$
$$\Box(u \wedge \neg\bigcirc d \rightarrow \bigcirc u) \tag{7}$$
$$\Box(d \wedge \neg\bigcirc u \rightarrow \bigcirc d) \tag{8}$$

In [8] it was shown how to use equivalence in the logic of THT as a sufficient condition for strong equivalence of two arbitrary TEL theories. The THT-equivalence test was performed by translating THT into LTL and using a model checker afterwards. This technique was applied on several examples to compare different alternative ASP representations of the same temporal scenario. Paradoxically, although this allowed an automated test to know whether two representations had the same behaviour, computing such a behaviour, that is, automatically obtaining the temporal equilibrium models of a given theory was still an open topic.

When compared to the ASP case, part of the difficulties for computing temporal equilibrium models came from the fact that the general syntax of TEL allows complete arbitrary nesting of connectives (that is, it coincides with general LTL), whereas ASP syntax is constrained to disjunctive logic programs, that is, rules with a *body* (the antecedent) with a conjunction of literals, and a *head* (the consequent) with a disjunction of atoms. In a recent work [9], it was shown that TEL could be reduced to a normal form closer to logic programs where, roughly speaking, in place of an atom p we can also use $\bigcirc p$, and any rule can be embraced by $\Box$. This normal form received the name of *Temporal Logic Programs* (TLPs) – for instance, (5)-(8) are TLP rules.

In this work we show how to compute the temporal equilibrium models for a subclass of TLPs called *splitable*. This subclass has a property we can call *future projected dependence* and that informally speaking can be described as "past does not depend on the future." Formally, this means that we cannot have rules where a head atom without $\bigcirc$ depends on a body atom with $\bigcirc$. In our example, (5)-(8) are also splitable TLP rules whereas, for instance, the following TLP rules are not in splitable form:

$$\Box(\neg\bigcirc p \rightarrow p) \tag{9}$$

$$\Box(\bigcirc p \rightarrow p) \tag{10}$$

since the truth of p "now" depends on p in the next situation $\bigcirc p$. This syntactic feature of splitable TLPs allows us to apply the so-called *splitting* technique [10] (hence the name of *splitable*) to our temporal programs. Informally speaking, splitting is applicable when we can divide an ASP program Π into a bottom Π_0 and a top Π_1 part, where Π_0 never depends on predicates defined in Π_1. If so, the stable models of Π can be computed by first obtaining the stable models of Π_0, and then using them to simplify Π_1 and compute the rest of information in a constructive way.

In the case of splitable TLPs, however, we cannot apply splitting by relying of multiple calls to an ASP solver since, rather than a single split between bottom and top part, we would actually have an *infinite* sequence of program "slices" $\Pi_0, \Pi_1, \Pi_2, \ldots$ where each Π_i depends on the previous ones. To solve this problem, we adopt a second ASP technique called *Loop Formulas* [11, 12], a set of formulas $LF(\Pi)$ for some program Π so that the stable models of the latter coincide with the classical models of $\Pi \cup LF(\Pi)$. In our case, $LF(\Pi)$ will contain formulas preceded by a $\Box$ operator, so that they affect to *all* program slices. As a result, the temporal equilibrium models of Π will correspond to the LTL models of $\Pi \cup LF(\Pi)$, which can be computed using an LTL model checker as a back-end.

The rest of the paper is organised as follows. In the next section, we recall the syntax and semantics of TEL. Section 3 describes the syntax of splitable TLPs and their relation to stable models. Next, in Section **??**, we explain how to construct the set of loop formulas $LF(\Pi)$ for any splitable TLP Π, proving also that the LTL models of $\Pi \cup LF(\Pi)$ are the temporal equilibrium models of Π. In Section **??** we comment a pair of examples and finally, Section **??** concludes the paper.

2 Preliminaries

Given a set of atoms At, a *formula* F is defined as in LTL following the grammar:

$$F ::= p \mid \bot \mid F_1 \wedge F_2 \mid F_1 \vee F_2 \mid F_1 \rightarrow F_2 \mid \bigcirc F \mid \Box F \mid \Diamond F$$

where $p \in At$. A *theory* is a finite set of formulas. We use the following derived operators[1] and notation:

$$\neg F \stackrel{\text{def}}{=} F \rightarrow \bot \qquad \bigcirc^i F \stackrel{\text{def}}{=} \bigcirc(\bigcirc^{i-1} F) \quad (\text{with } i > 1)$$
$$\top \stackrel{\text{def}}{=} \neg\bot \qquad \bigcirc^0 F \stackrel{\text{def}}{=} F$$
$$F \leftrightarrow G \stackrel{\text{def}}{=} (F \rightarrow G) \wedge (G \rightarrow F)$$

The semantics of the logic of *Temporal Here-and-There* (THT) is defined in terms of sequences of pairs of propositional interpretations. A (temporal) *interpretation* $\mathbf{M}$ is an infinite sequence of pairs $m_i = \langle H_i, T_i \rangle$ with $i = 0, 1, 2, \ldots$ where $H_i \subseteq T_i$ are sets of atoms standing for *here* and *there* respectively. For simplicity, given a temporal interpretation, we write $\mathbf{H}$ (resp. $\mathbf{T}$) to denote the sequence of pair components $H_0, H_1, \ldots$ (resp. $T_0, T_1, \ldots$). Using this notation, we will sometimes abbreviate the interpretation as $\mathbf{M} = \langle \mathbf{H}, \mathbf{T} \rangle$. An interpretation $\mathbf{M} = \langle \mathbf{H}, \mathbf{T} \rangle$ is said to be *total* when $\mathbf{H} = \mathbf{T}$.

Given an interpretation $\mathbf{M}$ and an integer number $k > 0$, by $(\mathbf{M}, k)$ we denote a new interpretation that results from "shifting" $\mathbf{M}$ in k positions, that is, the sequence of pairs $\langle H_k, T_k \rangle, \langle H_{k+1}, T_{k+1} \rangle, \langle H_{k+2}, T_{k+2} \rangle, \ldots$ Note that $(\mathbf{M}, 0) = \mathbf{M}$.

Definition 1 (satisfaction). *An interpretation* $\mathbf{M} = \langle \mathbf{H}, \mathbf{T} \rangle$ *satisfies a formula* φ*, written* $\mathbf{M} \models \varphi$*, when:*

1. $\mathbf{M} \models p$ *if* $p \in H_0$*, for any atom* p*.*
2. $\mathbf{M} \models \varphi \wedge \psi$ *if* $\mathbf{M} \models \varphi$ *and* $\mathbf{M} \models \psi$*.*
3. $\mathbf{M} \models \varphi \vee \psi$ *if* $\mathbf{M} \models \varphi$ *or* $\mathbf{M} \models \psi$*.*
4. $\langle \mathbf{H}, \mathbf{T} \rangle \models \varphi \rightarrow \psi$ *if* $\langle x, \mathbf{T} \rangle \not\models \varphi$ *or* $\langle x, \mathbf{T} \rangle \models \psi$ *for all* $x \in \{\mathbf{H}, \mathbf{T}\}$*.*
5. $\mathbf{M} \models \bigcirc\varphi$ *if* $(\mathbf{M}, 1) \models \varphi$*.*
6. $\mathbf{M} \models \Box\varphi$ *if* $\forall j \geq 0,\ (\mathbf{M}, j) \models \varphi$
7. $\mathbf{M} \models \Diamond\varphi$ *if* $\exists j \geq 0,\ (\mathbf{M}, j) \models \varphi$

A formula φ is *valid* if $\mathbf{M} \models \varphi$ for any $\mathbf{M}$. An interpretation $\mathbf{M}$ is a *model* of a theory Γ, written $\mathbf{M} \models \Gamma$, if $\mathbf{M} \models \alpha$, for all formula $\alpha \in \Gamma$.

We will make use of the following THT-valid equivalences:

$$\neg(F \wedge G) \leftrightarrow \neg F \vee \neg G \tag{11}$$
$$\neg(F \vee G) \leftrightarrow \neg F \wedge \neg G \tag{12}$$
$$\bigcirc(F \oplus G) \leftrightarrow \bigcirc F \oplus \bigcirc G \tag{13}$$
$$\bigcirc \otimes F \leftrightarrow \otimes \bigcirc F \tag{14}$$

for any binary connective $\oplus$ and any unary connective $\otimes$. This means that De Morgan laws (11),(12) are valid, and that we can always shift the $\bigcirc$ operator to all the operands of any connective.

[1] As shown in [9], the LTL binary operators $\mathcal{U}$ ("until") and $\mathcal{R}$ ("release") can be removed by introducing auxiliary atoms.

The logic of THT is an orthogonal combination of the logic of *Here-and-There* (HT) [13] and the (standard) linear temporal logic (LTL) [7]. On the one hand, HT is obtained by disregarding temporal operators, so that only the pair of sets of atoms $\langle H_0, T_0 \rangle$ is actually relevant and we use conditions 1-3 in Definition 1 for satisfaction of propositional theories. On the other hand, if we restrict the semantics to total interpretations, $\langle \mathbf{T}, \mathbf{T} \rangle \models \varphi$ corresponds to satisfaction of formulas $\mathbf{T} \models \varphi$ in LTL. This last correspondence allows rephrasing item 4 of Definition 1 as:

4'. $\langle \mathbf{H}, \mathbf{T} \rangle \models \varphi \rightarrow \psi$ if both (1) $\langle \mathbf{H}, \mathbf{T} \rangle \models \varphi$ implies $\langle \mathbf{H}, \mathbf{T} \rangle \models \psi$; and (2) $\mathbf{T} \models \varphi \rightarrow \psi$ in LTL.

Similarly $\langle \mathbf{H}, \mathbf{T} \rangle \models \varphi \leftrightarrow \psi$ if both (1) $\langle \mathbf{H}, \mathbf{T} \rangle \models \varphi$ iff $\langle \mathbf{H}, \mathbf{T} \rangle \models \psi$; and (2) $\mathbf{T} \models \varphi \leftrightarrow \psi$ in LTL. The following proposition can also be easily checked.

Proposition 1. *For any Γ and any $\mathbf{M} = \langle \mathbf{H}, \mathbf{T} \rangle$, if $\mathbf{M} \models \Gamma$ then $\mathbf{T} \models \Gamma$.* $\boxtimes$

We proceed now to define an ordering relation among THT models of a temporal theory, so that only the minimal ones will be *selected*. Given two interpretations $\mathbf{M} = \langle \mathbf{H}, \mathbf{T} \rangle$ and $\mathbf{M}' = \langle \mathbf{H}', \mathbf{T}' \rangle$ we say that $\mathbf{M}'$ is *lower or equal than* $\mathbf{M}$, written $\mathbf{M}' \leq \mathbf{M}$, when $\mathbf{T}' = \mathbf{T}$ and for all $i \geq 0$, $H'_i \subseteq H_i$. As usual, $\mathbf{M}' < \mathbf{M}$ stands for $\mathbf{M}' \leq \mathbf{M}$ but $\mathbf{M}' \neq \mathbf{M}$.

Definition 2 (Temporal Equilibrium Model). *An interpretation $\mathbf{M}$ is a* temporal equilibrium model *of a theory Γ if $\mathbf{M}$ is a total model of Γ and there is no other $\mathbf{M}' < \mathbf{M}$, $\mathbf{M}' \models \Gamma$.* $\boxtimes$

Note that any temporal equilibrium model is total, that is, it has the form $\langle \mathbf{T}, \mathbf{T} \rangle$ and so can be actually seen as an interpretation $\mathbf{T}$ in the standard LTL. *Temporal Equilibrium Logic* (TEL) is the logic induced by temporal equilibrium models.

When we restrict the syntax to ASP programs and HT interpretations $\langle H_0, T_0 \rangle$ we talk about (non-temporal) equilibrium models, which coincide with stable models in their most general definition [14].

3 Temporal Logic Programs

As we said in the introduction, in [9] it was shown that, by introducing auxiliary atoms, any temporal theory could be reduced to a normal form we proceed to describe. Given a signature At, we define a *temporal literal* as any expression in the set $\{p, \neg p, \bigcirc p, \neg \bigcirc p \mid p \in At\}$.

Definition 3 (Temporal rule). *A* temporal rule *is either:*

1. an initial rule *of the form*

$$B_1 \wedge \cdots \wedge B_n \rightarrow C_1 \vee \cdots \vee C_m \tag{15}$$

where all the B_i and C_j are temporal literals, $n \geq 0$ and $m \geq 0$.

2. *a* dynamic rule *of the form* $\Box r$*, where* r *is an initial rule.*
3. *a* fulfillment rule *like* $\Box(\Box p \to q)$ *or like* $\Box(p \to \Diamond q)$ *with* p, q *atoms.* ⊠

In the three cases, we respectively call rule *body* and rule *head* to the antecedent and consequent of the (unique) rule implication. In initial (resp.) dynamic rules, we may have an empty head $m = 0$ corresponding to $\bot$ – if so, we talk about an *initial* (resp. *dynamic*) *constraint.* A *temporal logic program*[2] (TLP for short) is a finite set of temporal rules. A TLP without temporal operators, that is, a set of initial rules without $\bigcirc$, is said to be an *ASP program*[3].

As an example of TLP take the program Π_1 consisting of:

$$\neg a \wedge \bigcirc b \to \bigcirc a \tag{16}$$

$$\Box(a \to b) \tag{17}$$

$$\Box(\neg b \to \bigcirc a) \tag{18}$$

where (16) is an initial rule and (17),(18) are dynamic rules.

Looking at the semantics of $\Box$ it seems clear that we can understand a dynamic rule $\Box r$ as an infinite sequence of expressions like $\bigcirc^i r$, one for each $i \geq 0$. Using (13),(14) we can shift $\bigcirc^i$ inside all connectives in r so that $\bigcirc^i r$ is equivalent to an initial rule resulting from prefixing any atom in r with $\bigcirc^i$. To put an example, if $r = (18)$ then $\bigcirc^2 r$ would correspond to $(\neg \bigcirc^2 b \to \bigcirc^3 a)$.

Definition 4 (i-expansion of a rule). *Given* $i \geq 0$*, the* i-expansion *of a dynamic rule* $\Box r$*, written* $(\Box r)^i$*, is a set of rules defined as:*

$$(\Box r)^i \begin{cases} \emptyset & \textit{if } i = 0 \textit{ and } r \textit{ contains some '}\bigcirc\textit{'} \\ \{\bigcirc^j r \mid 0 \leq j \leq i-1\} & \textit{if } i > 0 \textit{ and } r \textit{ contains some '}\bigcirc\textit{'} \\ \{\bigcirc^j r \mid 0 \leq j \leq i\} & \textit{otherwise} \end{cases}$$

If r *is an initial rule, its* i*-expansion is defined as:*

$$r^i \stackrel{\text{def}}{=} \begin{cases} \emptyset & \textit{if } i = 0 \textit{ and } r \textit{ contains some '}\bigcirc\textit{'} \\ r & \textit{otherwise} \end{cases}$$ ⊠

In this way, the superindex i refers to the longest sequence of $\bigcirc$'s used in the rule. For instance, $(18)^3$ would be:

$$\{ \ (\neg b \to \bigcirc a), \ \ (\neg \bigcirc b \to \bigcirc^2 a), \ \ (\neg \bigcirc^2 b \to \bigcirc^3 a) \ \}$$

We extend this notation to programs, so that given a TLP Π its i-expansion Π^i results from replacing each initial or dynamic rule r in Π by r^i. An interesting observation is that we can understand each Π^i as a (non-temporal) ASP program for signature $At^i \stackrel{\text{def}}{=} \{\text{“}\bigcirc^j p\text{”} \mid p \in At, 0 \leq j \leq i\}$ where we understand each “$\bigcirc^j p$” as a different propositional atom. This same notation can be applied

[2] In fact, as shown in [9], this normal form can be even more restrictive: initial rules can be replaced by atoms, and we can avoid the use of literals of the form $\neg\bigcirc p$.
[3] In ASP literature, this is called a a disjunctive program with negation in the head.

to interpretations. If $\mathbf{T}$ is an LTL interpretation (an infinite sequence of sets of atoms) for signature At its i-expansion would be the corresponding propositional interpretation for signature At^i defined as $\mathbf{T}^i \overset{\text{def}}{=} \{\bigcirc^i p \mid p \in T_i\}$ and if $\mathbf{M} = \langle \mathbf{H}, \mathbf{T} \rangle$ is a THT interpretation then its i-expansion is defined as the HT interpretation $\mathbf{M}^i \overset{\text{def}}{=} \langle \mathbf{H}^i, \mathbf{T}^i \rangle$. In all these cases, we also define the ω-*expansion* (or simply, *expansion*) as the infinite union of all i-expansions for all $i \geq 0$. Thus, for instance $(\Box r)^\omega \overset{\text{def}}{=} \bigcup_{i \geq 0} (\Box r)^i$ and similarly for Π^ω, At^ω, $\mathbf{T}^\omega$ and $\mathbf{M}^\omega$. It is interesting to note that, for any classical interpretation $\mathbf{T}'$ for signature At^ω, we can always build a corresponding LTL interpretation $\mathbf{T}$ in signature At such that $\mathbf{T}^\omega = \mathbf{T}'$. The following theorem establishes the correspondence between a temporal program and its expansion.

Theorem 1. *Let Π be a TLP without fulfillment rules. Then $\langle \mathbf{T}, \mathbf{T} \rangle$ is a temporal equilibrium model of Π under signature At iff $\mathbf{T}^\omega$ is a stable model of Π^ω under signature At^ω.* ⊠

The above theorem allows us reading a TLP with initial and dynamic rules as an ASP program with infinite "copies" of the same rule schemata. In many cases, this allows us to foresee the temporal equilibrium models of a TLP. For instance, if we look at our example TLP Π_2, it is easy to see that we should get $T_0 = \emptyset$ as the only rule affecting the situation $i = 0$ is $(17)^0 = (a \rightarrow b)$. For situation $i = 1$ we would have rules $(\bigcirc a \rightarrow \bigcirc b) \in (17)^1$ and $(\neg b \rightarrow \bigcirc a) \in (18)^1$ so that, given T_0 we obtain $\bigcirc a \wedge \bigcirc b$, that is, $T_1 = \{a, b\}$. For $i = 2$, the involved rules are $(\bigcirc^2 a \rightarrow \bigcirc^2 b) \in (17)^2$ and $(\neg \bigcirc b \rightarrow \bigcirc^2 a) \in (18)^2$ so that, given T_1 we obtain $T_2 = \emptyset$. In a similar way, for $i = 3$ we have rules $(\bigcirc^2 a \rightarrow \bigcirc^2 b)$ and $(\neg \bigcirc^2 b \rightarrow \bigcirc^3 a)$ leading to $T_3 = \{a, b\}$ and then this behaviour is repeated. To sum up, we get a unique temporal equilibrium model $\langle \mathbf{T}, \mathbf{T} \rangle$ for Π_2 where $\mathbf{T}$ can be captured by the regular expression $(\ \emptyset\ \{a, b\}\ )^+$.

In some cases, however, we may face new situations that are not common in standard ASP. For instance, consider the TLP Π_2 consisting of the two rules (9), (10). This program has no temporal equilibrium models. To see why, note that Π_2 is equivalent to $\Box(\neg \bigcirc p \vee \bigcirc p \rightarrow p)$ that, in its turn, is LTL-equivalent to $\Box p$. Thus, the only LTL-model $\mathbf{T}$ of Π_2 has the form $T_i = \{p\}$ for any $i \geq 0$. However, it is easy to see that the interpretation $\langle \mathbf{H}, \mathbf{T} \rangle$ with $H_i = \emptyset$ for all $i \geq 0$ is also a THT model, whereas $\mathbf{H} < \mathbf{T}$. Note that, by Theorem 1, this means that the ASP program Π_2^ω has no stable models, although it is an *acyclic program* and (finite) acyclic programs always have a stable model. The intuitive reason for this is that atoms $\bigcirc^i p$ infinitely depend on the future, and there is no way to build a founded reasoning starting from facts or the absence of them at a given end point[4].

[4] In fact, this example was extracted from a first-order counterpart, the pair of rules $\neg p(s(X)) \rightarrow p(X)$ and $p(s(X)) \rightarrow p(X)$, that were used in [15] to show that an acyclic program without a well-founded dependence ordering relation may have no stable models.

4 Splitting a Temporal Logic Program

Fortunately, most ASP programs dealing with transition systems represent rules so that past does not depend on the future. This is what we called *future projected dependence* and can be captured by the following subclass of TLPs.

Definition 5 (Splitable TLP). *A TLP Π for signature At is said to be* splitable *if Π consists of rules of any of the forms:*

$$B \wedge N \rightarrow H \tag{19}$$

$$B \wedge \bigcirc B' \wedge N \wedge \bigcirc N' \rightarrow \bigcirc H' \tag{20}$$

$$\Box(B \wedge \bigcirc B' \wedge N \wedge \bigcirc N' \rightarrow \bigcirc H') \tag{21}$$

where B and B' are conjunctions of atoms, N and N' are conjunctions of negative literals like $\neg p$ with $p \in At$, and H and H' are disjunctions of atoms. ⊠

The set of rules of form (19) in Π will be denoted $ini_0(\Pi)$ and correspond to initial rules for situation 0. The rules of form (20) in Π will be represented as $ini_1(\Pi)$ and are initial rules for the transition between situations 0 and 1. Finally, the set of rules of form (21) is written $dyn(\Pi)$ and contains dynamic rules. Both in (20) and (21), we understand that operator $\bigcirc$ is actually shifted until it only affects to atoms – this is always possible due to equivalences (13), (14). We will also use the formulas B, B', N, N', H and H' as sets, denoting the atoms that occur in each respective formula.

Notice that a rule of the form $\Box(B \wedge N \rightarrow H)$ (i.e., without $\bigcirc$ operator) is not splitable but can be transformed into the equivalent pair of rules $B \wedge N \rightarrow H$ and $\Box(\bigcirc B \wedge \bigcirc N \rightarrow \bigcirc H)$ which are both splitable. For instance, (17) becomes the pair of rules:

$$a \rightarrow b \tag{22}$$

$$\Box(\bigcirc a \rightarrow \bigcirc b) \tag{23}$$

As an example, Π_2=(9)-(10) is not splitable, whereas Π_1=(16),(18),(22),(23) is splitable being $ini_0(\Pi_1)$=(22), $ini_1(\Pi_1)$=(16) and $dyn(\Pi_1)$=(22),(18). In particular, in (16) we have the non-empty sets $B' = \{b\}$, $N = \{a\}$ and $H' = \{a\}$, whereas for (18) the sets are $N = \{b\}$, $H' = \{a\}$. It can be easily seen that the rules (5)-(8) are also in splitable form.

As we explained in the introduction, the most interesting feature of splitable TLPs is that we can apply the so-called *splitting* technique [10] to obtain their temporal equilibrium models in an incremental way. Let us briefly recall this technique for the case of ASP programs. Following [10] we define:

Definition 6 (Splitting set). *Let Π be an ASP program consisting of (non-temporal) rules like* (19)*. Then a set of atoms U is a* splitting set *for Π if, for any rule like* (19) *in Π: if $H \cap U \neq \emptyset$ then $(B \cup N \cup H) \subseteq U$. The set of rules satisfying $(B \cup N \cup H) \subseteq U$ are denoted as $b_U(\Pi)$ and called the* bottom *of Π with respect to U.* ⊠

Consider the program:

$$a \rightarrow c \tag{24}$$
$$b \rightarrow d \tag{25}$$
$$\neg b \rightarrow a \tag{26}$$
$$\neg a \rightarrow b \tag{27}$$

The set $U = \{a, b\}$ is a splitting set for Π being $b_U(\Pi) = \{(26), (27)\}$. The idea of splitting is that we can compute first each stable model X of $b_U(\Pi)$ and then use the truth values in X for simplifying the program $\Pi \setminus b_U(\Pi)$ from which the rest of truth values for atoms not in U can be obtained. Formally, given $X \subseteq U \subseteq At$ and an ASP program Π, for each rule r like (19) in Π such that $H \cap U \subseteq X$ and $N \cap U$ is disjoint from X, take the rule $\hat{r} : \hat{B} \wedge \hat{N} \rightarrow H$ where $\hat{B} = (B \setminus U)$ and $\hat{N} = (N \setminus U)$. The program consisting of all rules $\hat{r}$ obtained in this way is denoted as $e_U(\Pi, X)$. Note that this program is equivalent to replacing in all rules in Π each atom $p \in U$ by $\bot$ if $p \notin X$ and by $\top$ if $p \in X$.

In the previous example, the stable models of Π are $\{a\}$ and $\{b\}$. For the first stable model $X = \{a\}$, we get $e_U(\Pi \setminus b_U(\Pi), \{a\}) = \{\top \rightarrow c\}$ so that $X \cup \{c\} = \{a, c\}$ should be a stable model for the complete program Π. Similarly, for $X = \{b\}$ we get $e_U(\Pi \setminus b_U(\Pi), \{b\}) = \{\top \rightarrow d\}$ and a "completed" stable model $X \cup \{d\} = \{b, d\}$. The following result guarantees the correctness of this method in the general case.

Theorem 2 (from [10]). *Let U be a splitting set for a set of rules Π like* (19)*. A set of atoms X is an stable model of Π if, and only if both*

- *$X \cap U$ is a stable model of $b_U(\Pi)$*
- *$X \setminus U$ is a stable model of $e_U(\Pi \setminus b_U(\Pi), X \cap U)$.* ⊠

In [10] this result was generalised for an infinite sequence of splitting sets, showing an example of a logic program with variables and a function symbol, so that the ground program was infinite. We adapt next this splitting sequence result for the case of splitable TLPs in TEL.

From Definition 4 we can easily conclude that, when Π is a splitable TLP, its programs expansions have the form:

$$\Pi^0 = ini_0(\Pi)$$
$$\Pi^i = ini_0(\Pi) \cup ini_1(\Pi) \cup dyn(\Pi)^i \qquad \text{(for } i > 0\text{)}$$

Proposition 2. *Given a splitable TLP Π for signature At and any $i \geq 0$:*

(i) *At^i is a splitting set for Π^ω;*
(ii) *and $b_{At^i}(\Pi^\omega) = \Pi^i$.* ⊠

Given any rule like r like (20) of (21) and a set of atoms X, we define its *simplification* $simp(r, X)$ as:

$$simp(r, X) \stackrel{\text{def}}{=} \begin{cases} \bigcirc B' \wedge \bigcirc N' \rightarrow \bigcirc H' & \text{if } B \subseteq X \text{ and } N \cap X = \emptyset \\ \top & \text{otherwise} \end{cases}$$

Given some LTL interpretation $\mathbf{T}$, let us define now the sequence of programs:

$$\Pi[\mathbf{T}, i] \stackrel{\text{def}}{=} e_{At^i}\left(\Pi^\omega \setminus \Pi^i\ ,\ \mathbf{T}^i\right)$$

that is, $\Pi[\mathbf{T}, i]$ is the "simplification" of Π^ω by replacing atoms in At^i by their truth value with respect to $\mathbf{T}^i$. Then, we have:

Proposition 3.

$$\Pi[\mathbf{T}, 0] = (dyn(\Pi)^\omega \setminus dyn(\Pi)^1) \cup \{simp(r, T_0) \mid r \in ini_1(\Pi) \cup dyn(\Pi)\} \quad (28)$$

and, for any, $i \geq 1$

$$\Pi[\mathbf{T}, i] = (dyn(\Pi)^\omega \setminus dyn(\Pi)^{i+1}) \cup \ \{\bigcirc^i simp(r, T_i) \mid r \in dyn(\Pi)\} \quad (29)$$

⊠

As we can see, programs $\Pi[\mathbf{T}, i]$ maintain most part of $dyn(\Pi)^\omega$ and only differ in simplified rules. Let us call these sets of simplified rules:

$$\begin{aligned} slice(\Pi, \mathbf{T}, 0) &\stackrel{\text{def}}{=} \Pi^0 = ini_0(\Pi) \\ slice(\Pi, \mathbf{T}, 1) &\stackrel{\text{def}}{=} \{simp(r, T_0) \mid r \in ini_1(\Pi) \cup dyn(\Pi)\} \\ slice(\Pi, \mathbf{T}, i+1) &\stackrel{\text{def}}{=} \{\bigcirc^i simp(r, T_i) \mid r \in dyn(\Pi)\} \qquad \text{for } i \geq 1 \end{aligned}$$

Theorem 3 (Splitting Sequence Theorem). *Let $\langle \mathbf{T}, \mathbf{T} \rangle$ be a model of a splitable TLP Π. $\langle \mathbf{T}, \mathbf{T} \rangle$ is a temporal equilibrium model of Π iff*

(i) $\mathbf{T}^0 = T_0$ *is a stable model of* $slice(\Pi, \mathbf{T}, 0) = \Pi^0 = ini_0(\Pi)$ *and*
(ii) $(\mathbf{T}^1 \setminus At^0)$ *is a stable model of* $slice(\Pi, \mathbf{T}, 1)$ *and*
(iii) $(\mathbf{T}^i \setminus At^{i-1})$ *is a stable model of* $slice(\Pi, \mathbf{T}, i)$ *for* $i \geq 2$. ⊠

As an example, let us take again program $\Pi_1 = (16), (22), (23), (18)$. The program $\Pi_1^0 = ini_0(\Pi_1) = (22)$ has the stable model $\mathbf{T}^0 = \emptyset = T_0$. Then we take $slice(\Pi, \mathbf{T}, 1) = \{simp(r, T_0) \mid r \in ini_1(\Pi) \cup dyn(\Pi)\}$ that corresponds to:

$$\bigcirc b \rightarrow \bigcirc a \qquad \bigcirc a \rightarrow \bigcirc b \qquad \top \rightarrow \bigcirc a$$

whose stable model is $\{\bigcirc a, \bigcirc b\} = (\mathbf{T}^1 \setminus At^0)$ so that $T_1 = \{a, b\}$. In the next step, $slice(\Pi, \mathbf{T}, 2) = \{\bigcirc simp(r, T_1) \mid r \in dyn(\Pi)\} =$

$$\bigcirc^2 a \rightarrow \bigcirc^2 b \qquad \top$$

whose stable model is $\emptyset = (\mathbf{T}^2 \setminus At^1)$ so that $T_2 = \emptyset$. Then, we would go on with $slice(\Pi, \mathbf{T}, 3) = \{\bigcirc^2 simp(r, T_2) \mid r \in dyn(\Pi)\} =$

$$\bigcirc^3 a \rightarrow \bigcirc^3 b \qquad \top \rightarrow \bigcirc^3 a$$

leading to $\{\bigcirc^3 a, \bigcirc^3 b\}$ that is $T_3 = \{a, b\}$ and so on.

5 Loop formulas

Theorem 3 allows us building the temporal equilibrium models by considering an infinite sequence of finite ASP programs $\mathit{slice}(\Pi, \mathbf{T}, i)$. If we consider each program $\Pi' = \mathit{slice}(\Pi, \mathbf{T}, i+1)$ for signature $At^{i+1} \setminus At^i$ then, since it is a standard disjunctive ASP program, we can use the main result in [12] to compute its stable models by obtaining the classical models of a theory $\Pi' \cup LF(\Pi')$ where LF stands for *loop formulas*. To make the paper self-contained, we recall next some definitions and results from [12].

Given an ASP program Π we define its *(positive) dependency graph* $G(\Pi)$ where its vertices are At (the atoms in Π) and its edges are $E \subseteq At \times At$ so that $(p, p) \in E$ for any atom[5] p, and $(p, q) \in E$ if there is an ASP rule in Π like (19) with $p \in H$ and $q \in B$. A nonempty set L of atoms is called a *loop* of a program Π if, for every pair p, q of atoms in L, there exists a path from p to q in $G(\Pi)$ such that all vertices in the path belong to L. In other words, L is a loop of iff the subgraph of $G(\Pi)$ induced by L is strongly connected. Notice that reflexivity of $G(\Pi)$ implies that for any atom p, the singleton $\{p\}$ is also a loop.

Definition 7 (external support). *Given an ASP program Π for signature At, the* external support *formula of a set of atoms $Y \subseteq At$ with respect to Π, written $ES_\Pi(Y)$ is defined by:*

$$\bigvee_{r \in R(Y)} \left(B \wedge N \wedge \bigwedge_{p \in H \setminus Y} \neg p \right)$$

where $R(Y) = \{ r \in \Pi$ like (19) $\mid H \cap Y \neq \emptyset$ and $B \cap Y = \emptyset \}$. ⊠

Theorem 4 (from [12]). *Given a program Π for signature At, and a (classical) model $X \subseteq At$ of Π then X is a stable model of Π iff for every loop Y of Π, X satisfies $\bigvee_{p \in Y} p \rightarrow ES_\Pi(Y)$* ⊠

If we apply this result to each $s_i = \mathit{slice}(\Pi, \mathbf{T}, i)$ we obtain an infinite sequence of classical theories. Fortunately, these theories have a repetitive pattern and can be captured by a single, finite LTL theory.

Given a splitable TLP Π we will consider the dependency graph $G(\Pi)$ generated from the expanded (ASP) program Π^2, so that its nodes are atoms in the signature $At^2 = \{p, \bigcirc p, \bigcirc^2 p \mid p \in At\}$. We define the concept of loop $Y \subseteq At^2$ using this graph and signature. For instance, looking at $G(\Pi_1)$ in Figure 1 it is easy to see that loops for Π_1 are $\{\bigcirc a, \bigcirc b\}$ plus $\{A\}$ for any $A \in At^2$.

Theorem 5. *Let Π be a splitable TLP and $\mathbf{T}$ an LTL model of Π. Then $\langle \mathbf{T}, \mathbf{T} \rangle$ is a temporal equilibrium model of Π iff $\mathbf{T}$ is an LTL model of the union of*

[5] The original formulation in [12] did not consider reflexive edges, dealing instead with the idea of paths of length 0.

$$\begin{tikzcd}
a & \bigcirc a \arrow[d, bend right] & \bigcirc^2 a \\
b \arrow[u] & \bigcirc b \arrow[u, bend right] & \bigcirc^2 b \arrow[u]
\end{tikzcd}$$

Fig. 1. Graph $G(\Pi_1)$ (reflexive arcs are not displayed).

formulas $LF(Y)$ *defined as:*

$$\begin{array}{ll}
\bigvee_{p \in Y} p \rightarrow ES_{ini_0(\Pi)}(Y) & \textit{for any loop } Y \subseteq At^0 = At \\
\bigvee_{p \in Y} p \rightarrow ES_{ini_1(\Pi) \cup dyn(\Pi)^1}(Y) & \textit{for any loop } Y \subseteq (At^1 \setminus At^0) \\
\Box\Big(\bigvee_{p \in Y} p \rightarrow ES_{dyn(\Pi)^2 \setminus dyn(\Pi)^2}(Y) \Big) & \textit{for any loop } Y \subseteq (At^2 \setminus At^1) \quad \boxtimes
\end{array}$$

In our running example Π_1 we have $At^0 = \{a, b\}$ and $ini_0(\Pi) = (22)$ with two loops $\{a\}, \{b\}$ where $LF(\{a\}) = (a \rightarrow \bot)$ and $LF(\{b\}) = (b \rightarrow a)$. For $(At^1 \setminus At^0) = \{\bigcirc a, \bigcirc b\}$ we take the program $ini_1(\Pi_1) \cup dyn(\Pi)^1)$, that is, rules (16), (23), (18) ignoring $\Box$. We get three loops leading to the corresponding loop formulas $(\bigcirc a \rightarrow (\neg a \wedge \bigcirc b) \vee \neg b)$, $(\bigcirc b \rightarrow \bigcirc a)$ and $(\bigcirc a \vee \bigcirc b \rightarrow \neg b)$. Finally, for $At^2 \setminus At^1$ we have two loop formulas $\Box(\bigcirc^2 b \rightarrow \bigcirc^2 a)$ and $\Box(\bigcirc^2 a \rightarrow \neg \bigcirc b)$. It is not difficult to see that $\Pi_1 \cup LF(\Pi_1)$ is equivalent to the LTL theory: $\neg a \wedge \neg b \wedge \Box(\bigcirc a \leftrightarrow \neg b) \wedge \Box(\bigcirc b \leftrightarrow \neg b)$.

6 Conclusions

We have presented a class of temporal logic programs (that is ASP programs with temporal operators) for which their temporal equilibrium models (the analogous to stable models) can be computed by translation to LTL. To this aim, we have combined two well-known techniques in the ASP literature called splitting and loop formulas. This syntactic class has as restriction so that rule heads never refer to a temporally previous situation than those referred in the body. Still, it is expressive enough to cover most ASP examples dealing with dynamic scenarios.

We have implemented a system, called `STeLP` that uses this technique to translate a program and calls an LTL model checker to obtain its temporal equilibrium models, in the form of a Büchi automaton. This tool allows some practical features like dealing with variables or specifying fluents and actions[6].

References

1. Niemelä, I.: Logic programs with stable model semantics as a constraint programming paradigm. Annals of Mathematics and Artificial Intelligence **25** (1999) 241–273

[6] More examples and information can be obtained in `http://www.dc.fi.udc.es/~cabalar/stelp.pdf`

2. Marek, V., Truszczyński, M. In: Stable models and an alternative logic programming paradigm. Springer-Verlag (1999) 169–181
3. Gelfond, M., Lifschitz, V.: The stable model semantics for logic programming. In Kowalski, R.A., Bowen, K.A., eds.: Logic Programming: Proc. of the Fifth International Conference and Symposium (Volume 2). MIT Press, Cambridge, MA (1988) 1070–1080
4. Cabalar, P., Vega, G.P.: Temporal equilibrium logic: a first approach. In: Proc. of the 11th International Conference on Computer Aided Systems Theory, (EUROCAST'07). LNCS (4739). (2007) 241–248
5. Pearce, D.: A new logical characterisation of stable models and answer sets. In: Non monotonic extensions of logic programming. Proc. NMELP'96. (LNAI 1216). Springer-Verlag (1996)
6. Pearce, D.: Equilibrium logic. Annals of Mathematics and Artificial Intelligence **47**(1-2) (2006) 3–41
7. Manna, Z., Pnueli, A.: The Temporal Logic of Reactive and Concurrent Systems: Specification. Springer-Verlag (1991)
8. Aguado, F., Cabalar, P., Pérez, G., Vidal, C.: Strongly equivalent temporal logic programs. In: Proc. of the 11th European Conference on Logics in Artificial Intelligence (JELIA'08). LNCS (vol. 5293). (2008) 8–20
9. Cabalar, P.: A normal form for linear temporal equilibrium logic. In: Proc. of the 12th European Conference on Logics in Artificial Intelligence (JELIA'10). Lecture Notes in Computer Science. Volume 2258. Springer-Verlag (2010) 64–76
10. Lifschitz, V., Turner, H.: Splitting a logic program. In: Proceedings of the 11th International Conference on Logic programming (ICLP'94). (1994) 23–37
11. Lin, F., Zhao, Y.: ASSAT: Computing answer sets of a logic program by SAT solvers. In: Artificial Intelligence. (2002) 112–117
12. Ferraris, P., Lee, J., Lifschitz, V.: A generalization of the Lin-Zhao theorem. Annals of Mathematics and Artificial Intelligence **47** (2006) 79–101
13. Heyting, A.: Die formalen Regeln der intuitionistischen Logik. Sitzungsberichte der Preussischen Akademie der Wissenschaften, Physikalisch-mathematische Klasse (1930) 42–56
14. Ferraris, P.: Answer sets for propositional theories. In: Proc. of the 8th International Conference on Logic Programming and Nonmonotonic Reasoning (LPNMR'05). LNCS (vol. 3662). (2005) 119–131
15. Fages, F.: Consistency of Clark's completion and existence of stable models. Methods of Logic in Computer Science **1** (1994) 51–60

Appendix. Proofs

Lemma 1. *Let r be an initial rule like* (15)*, and* $\mathbf{M}$ *a THT interpretation. Then* $(\mathbf{M}, i) \models r$ *iff* $\mathbf{M}^\omega \models \bigcirc^i r$.

Proof. It directly follows from construction of $\mathbf{M}^\omega$ and simple inspection of the THT satisfaction relation. $\boxtimes$

Lemma 2. *Let Π be a TLP without fulfillment rules. Then* $\langle \mathbf{H}, \mathbf{T} \rangle \models \Pi$ *in THT iff* $\langle \mathbf{H}^\omega, \mathbf{T}^\omega \rangle \models \Pi^\omega$ *in HT.*

Proof. We prove that for any rule r in Π, $\mathbf{M} = \langle \mathbf{H}, \mathbf{T} \rangle \models r$ iff $\mathbf{M}^\omega = \langle \mathbf{H}^\omega, \mathbf{T}^\omega \rangle \models r^\omega$. We have two cases:

1. Let r be an initial rule r. The condition $\mathbf{M} \models r$ can be expressed as $(\mathbf{M}, 0) \models r$ and, from Lemma 1, this is equivalent to $\mathbf{M}^\omega \models \bigcirc^0 r$ that is $\mathbf{M}^\omega \models r$.
2. Let r be a dynamic rule $r = \Box s$. The condition $\mathbf{M} \models \Box s$ is equivalent to: $\forall i \geq 0, (\mathbf{M}, i) \models s$. From Lemma 1, this is equivalent to $\forall i \geq 0, \mathbf{M}^\omega \models \bigcirc^i s$, that is, $\mathbf{M}^\omega \models (\Box s)^\omega$. $\boxtimes$

Proof of Theorem 1.. For the left to right direction, suppose $\langle \mathbf{T}, \mathbf{T} \rangle$ is a temporal equilibrium model of Π. This implies $\langle \mathbf{T}, \mathbf{T} \rangle \models \Pi$ and, by Lemma 2, we get $\langle \mathbf{T}^\omega, \mathbf{T}^\omega \rangle \models \Pi^*$. Assume $\mathbf{T}^\omega$ is not a stable model of Π^ω. This means that $\langle \mathbf{T}^\omega, \mathbf{T}^\omega \rangle$ is not an equilibrium model of that program, that is, there exists some smaller $\mathbf{H}^\omega \subset \mathbf{T}^\omega$ such that $\langle \mathbf{H}^\omega, \mathbf{T}^\omega \rangle \models \Pi^\omega$. But by Lemma 2 again we would have $\langle \mathbf{H}, \mathbf{T} \rangle \models \Pi$ and as $\mathbf{H} < \mathbf{T}$ (since $\mathbf{H}^\omega \subset \mathbf{T}^\omega$) we conclude that $\langle \mathbf{T}, \mathbf{T} \rangle$ cannot be a temporal equilibrium model of Π, reaching a contradiction.

For the right to left direction, take any stable model $\mathbf{T}^\omega$ of Π^ω. Since $\mathbf{T}^\omega \models \Pi^\omega$ or equivalently $\langle \mathbf{T}^\omega, \mathbf{T}^\omega \rangle \models \Pi^\omega$ in HT, from Lemma 2, we get $\langle \mathbf{T}, \mathbf{T} \rangle \models \Pi$. Suppose that for some $\mathbf{H} \subset \mathbf{T}$, $\langle \mathbf{H}, \mathbf{T} \rangle \models \Pi$. By Lemma 2 we obtain $\langle \mathbf{H}^\omega, \mathbf{T}^\omega \rangle \models \Pi^\omega$ and as $\mathbf{H}^\omega \subset \mathbf{T}^\omega$ (since $\mathbf{H} \subset \mathbf{T}$) we obtain that $\langle \mathbf{T}^\omega, \mathbf{T}^\omega \rangle$ cannot be an equilibrium model for Π^ω, reaching a contradiction. $\boxtimes$

Proof of Proposition 2. (i) If a rule in Π^ω has some head atom in At^i it will have the form $\bigcirc^j p$ for some $0 \leq j \leq i$. By construction, in any of the three rule types (19),(20),(21) the whole head will consist of (expanded) atoms of the form $\bigcirc^j q$ whereas the body may contain atoms of the form $\bigcirc^j q$ or $\bigcirc^{j-1} q$. All these expanded atoms have a sequence of $\bigcirc$'s of length smaller than or equal to i, so they also belong to At^i.
(ii) By Definition 4 the longest sequence of $\bigcirc$'s occurring in any r^i (and so, in Π^i) will have length i. This implies that $\Pi^i \subseteq b_{At^i}(\Pi^\omega)$. To prove the opposite $b_{At^i}(\Pi^\omega) \subseteq \Pi^i$ suppose we had $r \in b_{At^i}(\Pi^\omega)$ but $r \notin \Pi^i$. Since $b_{At^i}(\Pi^\omega) \subseteq \Pi^\omega$, we get $r \in \Pi^\omega \setminus \Pi^i$. Then, for some $j > i$, $r \in \Pi^j \setminus \Pi^i$. Let us take the smallest j satisfying this. Then, $r \in \Pi^j$ but $r \notin \Pi^{j-1}$. But then, by construction of the program expansion, r must contain some atom of the form $\bigcirc^j p$. Finally, as $j > i$, $\bigcirc^j p \notin At^i$ and so $r \notin b_{At^i}(\Pi^\omega)$ reaching a contradiction. $\boxtimes$

Proof of Proposition 3. First note that $\Pi^\omega \setminus \Pi^0 = ini_1(\Pi) \cup dyn(\Pi)^\omega$. We obtain (28) by calculating $e_{At^0}\left(\Pi^\omega \setminus \Pi^0\ ,\ \mathbf{T}^0\right)$, since $\mathbf{T}^0 = T_0$. Now take $i \geq 1$ and suppose that (29) is true. Then

$$\begin{aligned}\Pi[\mathbf{T}, i+1] &= e_{At^{i+1}}\left(\Pi^\omega \setminus \Pi^{i+1}\ ,\ \mathbf{T}^{i+1}\right)\\ &= \left(\Pi^\omega \setminus \Pi^{i+2}\right) \bigcup e_{At^{i+1}}\left(\Pi^{i+2} \setminus \Pi^{i+1}\ ,\ \mathbf{T}^{i+1}\right)\end{aligned}$$

and since $i+2 > 1$ this is the same than:

$$\begin{aligned}=&\left(dyn(\Pi)^\omega \setminus dyn(\Pi)^{i+2}\right) \bigcup e_{At^{i+1}}\left(\Pi^{i+2} \setminus \Pi^{i+1}\ ,\ \mathbf{T}^{i+1}\right)\\ =&\left(dyn(\Pi)^\omega \setminus dyn(\Pi)^{i+1}\right)\\ &\bigcup\ \{\bigcirc^{i+2}B' \wedge \bigcirc^{i+2}N' \rightarrow \bigcirc^{i+2}H'\\ &\qquad \mid \text{rule like (21)} \in dyn(\Pi) \text{ and } \ B \subseteq T_{i+1}, N \cap T_{i+1} = \emptyset\}\end{aligned}$$

⊠

Proof of Theorem 3. Using Theorem 1, we have to prove that $\mathbf{T}^\omega$ is a stable model of Π^ω iff the two above conditions are satisfied. As At^0 is a splitting set of Π^ω, $\mathbf{T}^\omega$ is an stable model of P^ω iff

- $\mathbf{T}^\omega \cap At^0 = T_0$ is a stable model of $b_{At^0}(\Pi^\omega) = \Pi^0$, which is (i), plus
- $\mathbf{T}^\omega \setminus At^0$ is a stable model of $e_{At^0}(\Pi^\omega \setminus \Pi^0, \mathbf{T}^0) = \Pi[\mathbf{T}, 0]$.

Since, At^1 is again a splitting set of $\Pi[\mathbf{T}, 0]$, the last condition is equivalent to:

- $(\mathbf{T}^\omega \setminus At^0) \cap At^1 = (\mathbf{T}^1 \setminus At^0)$ is a stable model of $b_{At^1}(\Pi[\mathbf{T}, 0])$ that corresponds to $\{simp(r, T_0) \mid r \in ini_1(\Pi) \cup dyn(\Pi)\}$, so this is (ii),
- and $(\mathbf{T}^\omega \setminus At^0) \setminus At^1 = (\mathbf{T}^\omega \setminus At^1)$ is a stable model of $e_{At^1}(\Pi^\omega \setminus \Pi^1, \mathbf{T}^1) = \Pi[\mathbf{T}, 1]$.

As At^2 is a splitting set for $\Pi[\mathbf{T}, 1]$ we can apply a similar reasoning to conclude that the last item above is equivalent to:

- $(\mathbf{T}^\omega \setminus At^1) \cap At^2 = (\mathbf{T}^2 \setminus At^1)$ is a stable model of $b_{At^2}(\Pi[\mathbf{T}, 1])$ that corresponds to $\{\bigcirc simp(r, T_1) \mid r \in dyn(\Pi)\}$, so this is (iii) with $i = 2$, the base case,
- and $(\mathbf{T}^\omega \setminus At^1) \setminus At^2 = (\mathbf{T}^\omega \setminus At^2)$ is a stable model of $e_{At^2}(\Pi^\omega \setminus \Pi^2, \mathbf{T}^2) = \Pi[\mathbf{T}, 2]$.

and the induction step for (iii) follows by replacing in the two conditions above $1, 2$ respectively by i and $i+1$. ⊠

Proof of Theorem 5. For a proof sketch, notice that there cannot be loops between two different situations since this would require that some past situation depended on the future. So, each loop is always inside some $slice(\Pi, \mathbf{T}, i)$ and all atoms for At^{i-1} are external to the loop. Since the first two slices are affected by initial rules, they are specified in a separate case, whereas from slice 2 on we get a repetitive pattern using $\Box$. ⊠