Temporal Answer Sets

Pedro Cabalar

Dept. of Computer Science,
University of Corunna, Galicia, Spain.
e-mail: cabalar@dc.fi.udc.es

Abstract

In this paper, we propose an extension of disjunctive extended logic programming (under the answer sets semantics) for dealing with discrete linear time using modal temporal logic. We outline the advantages of using such formalism for representing common problems in reasoning about actions, specially those based on narrative approaches. In a first part, we present a simplified three-valued reformulation of the answer sets semantics. After that, we apply the same modal linear-time extension than that of TEMPLOG, a temporal logic programming formalism based on the least Herbrand model semantics. We show that the result subsumes TEMPLOG, generalizing it for any syntax-independent program with disjunction and both types of negation, default and classical. Finally, as an example of use, we present a translation of an action description language into this new formalism.

Keywords: Answer Sets, Temporal Logic Programming, TEMPLOG, Narrative-Based Action Theories.

1 Introduction

Since the introduction of declarative semantics for logic programs, first with *stable models* [12] and later with other approaches like *well-founded semantics* [30], a strong connection between logic programming and general purpose nonmonotonic logics has been established. This has led to the common use of logic programming as a knowledge representation tool, which, in its turn, has also meant the addition of new expressive features like disjunction or classical negation. For instance, stable models have been generalized into *answer sets* [13] while similar steps have been followed for well-founded semantics (for a recent overview, see for instance [8]).

In [14], answer sets are proposed as a low level formalism for translating high level languages for reasoning about actions. Although many action description languages have followed this methodology [14, 2, 29, 19], all their translations require representing situations in an explicit way, defining, for instance, in the case of narrative approaches, a sort of non-negative integers. Since *the presence of temporal structures is constant* in all the logic programming representations of action theories, we study how to incorporate them into the basic logic programming framework, as a similar extension to those previously done for classical negation or disjunction.

In this paper, we extend disjunctive extended logic programming with linear-time modal operators. We follow the same modal approach than in TEMPLOG [1], a temporal logic programming language with a least Herbrand model semantic characterization [3]. Modal logic constitutes a simple and natural way of expressing the temporal properties of the domain without needing the introduction of explicit timepoints or integer arithmetics. This will make easier the future work for obtaining decidability results or, as happens in the TEMPLOG case, for pointing out how standard resolution procedures (like SLD) can be specialized for dealing with temporal structures (*temporal* SLD).

For its use in knowledge representation, however, TEMPLOG has a strong limitation: it does not allow negation as failure. This restriction is too strong, since it *makes difficult the representation of the commonsense law of inertia* necessary for solving the *frame problem* [22]. In this way, our work can be also seen as a generalization of TEMPLOG for dealing with classical and default negation, becoming a starting point for extending temporal SLD for negation as failure.

The paper is organized as follows. In the next section, we propose a three-valued reformulation of the answer sets semantics, called NF_3 which simplifies the introduction of the temporal extension, TNF_3, presented in section 3. Next, as an example of use, section 4 contains a translation of the action description language $\mathcal{B}$ [15]. In section 5, we include the proof of subsumption of TEMPLOG programs. Finally, the last section makes a comparison to related work and concludes the paper.

2 Three-valued negation as failure, NF_3

As explained before, for making an easier extension to the temporal case, we will reformulate answer sets so that we can avoid the Gelfond and Lifschitz's *modulo* syntactic transformation [13]. As we will see in section 6, the approach presented here, NF_3, is directly inspired to the encoding of answer sets into *Minimal Belief and Negation as Failure* (MBNF) [18], and can be also related to the more recent reformulation of answer sets presented in [26].

Intuitively, NF_3 uses three-valued logic for reasoning about which literals are believed or not. To represent this, we include Lukasiewicz's operator[1] $\boldsymbol{l}$, so that a formula $\boldsymbol{l}\phi$ is never unknown and points out if ϕ is valuated to true or not. When ϕ is a literal, we can read $\boldsymbol{l}\phi$ as "ϕ is believed". We will also add another new operator, $\boldsymbol{d}$ for handling two separated sets of literals: the *actual* world I^a and the the *dual* world, I^d. This dual world will be used for fixing the default part of a given theory.

Formally, NF_3 syntax and semantics are defined as follows. We begin with a propositional signature or set of atoms. Theories with predicates and variables are treated as the set of all their ground instances. Let ϕ, ψ be NF_3 formulas and P an atom. The following expressions are also NF_3 formulas: P, $\neg\phi$, $\phi \vee \psi$, $\boldsymbol{l}\phi$, $\boldsymbol{d}\phi$.

[1] See for instance formulation presented in [5], pag. 8, where the $\boldsymbol{l}$ operator is denoted as $\Box$.

Usual derived operators ($\wedge, \supset, \equiv$) and propositional constants $\top$ and $\bot \stackrel{\text{def}}{=} \neg\top$, are also defined.

A *literal* is any atom or its negation. We say that a set of literals is *consistent* if, for any atom P, it does not include both literals P and $\neg P$, and we say that it is *maximal* (or *complete*) when it contains a literal for each atom in the signature. A set of literals is said to be *logically closed* if either it is consistent or it is the set of all possible literals. An *interpretation*, M, is defined as a triple (x, I^a, I^d), of logically closed sets of literals, being $x \in \{I^a, I^d\}$ the world which is currently being used (either the actual or the dual world).

Definition 1 *The* valuation *of a formula ϕ with respect to interpretation $M = (x, I^a, I^d)$, denoted as ϕ^x (we omit the fixed components I^a and I^d), is a value among $\{0, 1/2, 1\}$ (*false*, *unknown *or* true*, respectively), assigned following the next conditions:*

1) $P^x = \begin{cases} 1 & \text{if } P \in x \\ 0 & \text{if } \neg P \in x \\ 1/2 & \text{otherwise} \end{cases}$

2) $(\neg\phi)^x = 1 - \phi^x$

3) $(\phi \vee \psi)^x = max(\phi^x, \psi^x)$

4) $(\top)^x = 1$

5) $(\boldsymbol{l}\phi)^x = \begin{cases} 1 & \text{if } \phi^x = 1 \\ 0 & \text{otherwise} \end{cases}$

6) $(\boldsymbol{d}\phi)^x = \phi^{I^d}$

An interpretation (I^a, I^a, I^d) *satisfies* a formula ϕ when it makes it true, i.e., $\phi^{I^a} = 1$. We say that an interpretation is a *model* of a theory T when it satisfies all its formulas, and that a theory T *entails* a formula ϕ, denoted as $T \models \phi$, if all the models of T satisfy ϕ. As we can see, the scope of the $\boldsymbol{d}$ operator determines in which world x a subformula is valuated. Initially, x is the actual world I^a, but when a $\boldsymbol{d}$ occurs, its subformula is valuated in the dual world I^d.

Let $\phi \Leftrightarrow \psi$ represent semantic equivalence, that is, for any interpretation $M = (x, I^a, I^d)$, $\phi^x = \psi^x$. The following properties for NF_3 can be easily proved:

P1. $\boldsymbol{l}(\phi \wedge \psi) \Leftrightarrow \boldsymbol{l}\phi \wedge \boldsymbol{l}\psi$
P2. $\boldsymbol{l}(\phi \vee \psi) \Leftrightarrow \boldsymbol{l}\phi \vee \boldsymbol{l}\psi$
P3. $\boldsymbol{l}\boldsymbol{l}\phi \Leftrightarrow \boldsymbol{l}\phi$
P4. $\boldsymbol{l}\neg\boldsymbol{l}\phi \Leftrightarrow \neg\boldsymbol{l}\phi$
P5. $\boldsymbol{d}\neg\phi \Leftrightarrow \neg\boldsymbol{d}\phi$
P6. $\boldsymbol{d}(\phi \wedge \psi) \Leftrightarrow \boldsymbol{d}\phi \wedge \boldsymbol{d}\psi$
P7. $\boldsymbol{d}(\phi \vee \psi) \Leftrightarrow \boldsymbol{d}\phi \vee \boldsymbol{d}\psi$
P8. $\boldsymbol{d}\boldsymbol{l}\phi \Leftrightarrow \boldsymbol{l}\boldsymbol{d}\phi$
P9. $\boldsymbol{d}\boldsymbol{d}\phi \Leftrightarrow \boldsymbol{d}\phi$

Notice that, though we allow nested occurrences of $\boldsymbol{d}$ operator, any theory can, in fact, be transformed into an equivalent one with $\boldsymbol{d}$ operator only applied to atoms (it is enough with recursively applying properties from P5 to P9).

Within this basic monotonic framework, we will impose a selection criterion on models that will make it to become a nonmonotonic formalism. Given a theory T and a set of literals I, we define the function $min_T(I)$ as the minimal (with respect to set inclusion) logically closed set of literals I' from those (I', I', I) models of T.

Definition 2 (Selected model) *Given a* NF$_3$ *theory* T*, a model with shape* (I, I, I)*, is said to be a* selected model *iff* $I = min_T(I)$.

For simplicity sake we will represent selected models directly as (I) instead of (I, I, I). It can be easily seen that the entailment relation induced by selected models, $\approx$, becomes nonmonotonic. For instance, think about theory $T_1 = \{\neg \boldsymbol{l}\,\boldsymbol{d}P \supset \boldsymbol{l}\neg P\}$. The models of T_1 are those (I^a, I^a, I^d) where:

	I^a	I^d
M_1	$\{P\}$	$\{P\}$
M_2	$\{\neg P\}$	$\{P\}$
M_3	$\emptyset$	$\{P\}$
M_4	$\{\neg P\}$	$\{\neg P\}$
M_5	$\{\neg P\}$	$\emptyset$

Selected models must have the shape $I^a = I^d$. We have two candidates, M_1 and M_4. Fixing $I^d = \{P\}$, the minimal I^a is $\emptyset$, that is, $min_T(\{P\}) = \emptyset$, and so M_1 is not a selected model. The only fixpoint is $min_T(\{\neg P\}) = \{\neg P\}$, and so, M_4 is the unique selected model, which means that $T_1 \approx \neg P$. However, for the theory $T_2 = T_1 \cup \{P\}$ we have $T_2 \not\approx \neg P$. Its unique model, which is also selected model, is M_1.

We will show now that NF$_3$ selected models can be actually used as an alternative definition of answer sets for Disjunctive Extended Logic Programs (DELP). To this aim, we define in NF$_3$ the following derived operators:

$$
\begin{aligned}
\psi \leftarrow \phi &\stackrel{\text{def}}{=} \boldsymbol{l}\phi \supset \boldsymbol{l}\psi \\
not\ \phi &\stackrel{\text{def}}{=} \neg \boldsymbol{l}\,\boldsymbol{d}\phi \\
\phi, \psi &\stackrel{\text{def}}{=} \phi \wedge \psi \\
\phi | \psi &\stackrel{\text{def}}{=} \phi \vee \psi
\end{aligned}
$$

so that a general DELP rule:

$$\gamma_1 | \ldots | \gamma_p \leftarrow \alpha_1, \ldots, \alpha_n, not\ \beta_1, \ldots, not\ \beta_m$$

becomes the NF$_3$ formula:

$$\boldsymbol{l}(\alpha_1 \wedge \ldots \wedge \alpha_n \wedge \neg \boldsymbol{l}\,\boldsymbol{d}\beta_1 \wedge \ldots \wedge \neg \boldsymbol{l}\,\boldsymbol{d}\beta_m) \supset \boldsymbol{l}(\gamma_1 \vee \ldots \vee \gamma_p)$$

which in its turn, applying properties P1 and P4 to the antecedent and P2 to the consequent, amounts to:

$$(\boldsymbol{l}\alpha_1 \wedge \ldots \wedge \boldsymbol{l}\alpha_n \wedge \neg \boldsymbol{l}\,\boldsymbol{d}\beta_1 \wedge \ldots \wedge \neg \boldsymbol{l}\,\boldsymbol{d}\beta_m) \supset (\boldsymbol{l}\gamma_1 \vee \ldots \vee \boldsymbol{l}\gamma_p)$$

Proposition 1 *A consistent set of literals I is an answer set of a program Π iff I is a* NF_3 *selected model of* Π.

Proof First, we begin showing that, models of the reduct Π^I correspond to those selected models of Π that have the shape (I', I', I), that is, those ones fixing I. If we fix I, we obtain that, if $P \in I$, *not* P will always be valuated to false, whereas if $P \notin I$, *not* P will always be true. Thus, when I is fixed, we can substitute all the *not* P literals by their fixed truth values. After this substitution, the resulting program is trivially equivalent to Π^I, which is a program without default negation. Second, an answer set for a program like Π^I is defined as the minimal set being: (1) logically closed and (2) closed with respect to Π^I. The first condition is already required in the definition of NF_3 interpretation. The second one is also satisfied since, requiring the rules to be true under NF_3 semantics is a rephrasing of the definition of being closed with respect to a program. ∎

3 Temporal negation as failure: TNF_3

The answer sets reformulation done with NF_3 allows us to define a natural temporal extension, called TNF_3, where we introduce two new binary modal operators: $\bigcirc$ and $\Box$, standing for "in the next situation" and "from now on" respectively. We also define the usual derived operator $\Diamond\phi \stackrel{\text{def}}{=} \neg\Box\neg\phi$, standing for "eventually" or "at some future moment."

A *narrative* ν is defined as a pair (M, j), where $M = (x, I^a, I^d)$, j is a nonnegative integer, $x \in \{I^a, I^d\}$, and I^a, I^d are (possibly infinite) sequences of logically closed sets of literals ($I^a = \{I^a_0, I^a_1, \ldots\}, I^d = \{I^d_0, I^d_1, \ldots\}$).

Definition 3 (Valuation of a formula) *Given a narrative (M, j), $M = (x, I^a, I^d)$, its associated truth valuation is an extension of Definition 1, replacing x by x_j, I^d by I^d_j and adding the interpretations for the new operators:*

7) $(\bigcirc\phi)^{x_j} = \phi^{x_{j+1}}$

8) $$(\Box\phi)^{x_j} = \begin{cases} 1 & \text{if } \phi^{x_k} = 1 \textit{ for all } k \geq j \\ 0 & \text{if } \phi^{x_k} = 0 \textit{ for some } k \geq j \\ 1/2 & \text{otherwise} \end{cases}$$

A sequence of sets of literals I is said to be logically closed if either all the $I_k \in I$ are consistent or all the $I_k \in I$ are equal to the set of all literals. We say that a sequence of sets of literals I' is included into a another sequence I, $I' \sqsubseteq I$ iff for any position k, $I'_k \subseteq I_k$. Function min_T is then extended for sequences so that $I' = min_T(I)$ iff I' is the minimal (with respect to $\sqsubseteq$ relation) logically closed sequence from those $((I', I', I), 0)$ models of T.

Definition 4 Temporal Answer Set *A narrative with shape $((I, I, I), 0)$ is said to be a selected narrative or* temporal answer set *of a* TNF_3 *theory T iff $I = min_T(I)$.*

We again use abbreviation (I) for $((I, I, I), 0)$. Note that any narrative $((x, I^a, I^d), 0)$ can be alternatively seen as a NF_3 interpretation (x, I^a, I^d) for an infinite signature in which, for each proposition P, we consider the atoms $P, \bigcirc P, \bigcirc^2 P, \bigcirc^3 P, \ldots$, standing for P at situation $0, 1, 2, 3, \ldots$ respectively.

Let us see an example formulating the well known Yale Shooting scenario [16]. Let the theory T_{YS} be:

$$\Box(\bigcirc \neg Alive \leftarrow Shoot, Loaded) \tag{1}$$
$$\Box(\bigcirc \neg Loaded \leftarrow Shoot, Loaded) \tag{2}$$
$$\Box(\bigcirc Loaded \leftarrow Load) \tag{3}$$
$$\neg Loaded, Alive, Load, \bigcirc Wait, \bigcirc\bigcirc Shoot \tag{4}$$

together with the inertia law schemata:

$$\Box(\bigcirc p \quad \leftarrow \quad not\ \bigcirc \neg p, p) \tag{5}$$
$$\Box(\bigcirc \neg p \quad \leftarrow \quad not\ \bigcirc p, \neg p) \tag{6}$$

for p varying in $\{Alive, Loaded\}$.

Proposition 2 *The unique temporal answer set of theory T_{YS} is (I), where:*

$$\begin{aligned} I_0 &= \{Alive, \neg Loaded, Load\} \\ I_1 &= \{Alive, Loaded, Wait\} \\ I_2 &= \{Alive, Loaded, Shoot\} \\ I_3 &= \{\neg Alive, \neg Loaded\} \\ I_{j+1} &= I_j \textit{ for any } j \geq 3 \end{aligned}$$

Proof Direct facts (4) explain the initial situation I_0 and the occurrence of all the actions $Load, Shoot$ and $Wait$. Besides, rule (3) also fixes the occurrence of $Loaded$ in I_1. We will prove now that literal $Alive$ must be present in I_1. Let us assume that $\neg Alive \not\in I_1$. Then, after interpreting default negation, we would obtain that rule (5) for $Alive$ at I_0 becomes applicable and so, we obtain $\bigcirc Alive$, i.e., $Alive \in I_1$. If $\neg Alive \in I_1$, then we do not obtain any rule applicable for $\bigcirc \neg Alive$ and so, there would exist a lower model (I^a, I^a, I), in which $\neg Alive \not\in I_1^a$. This same reasoning can be applied for explaining $Alive$ and $Loaded$ in I_2. Once I_2 is fixed, situation I_3 is then uniquely established by application of rules (1) and (2). Finally, we would repeat again the reasoning for $Alive$ at I_1 to explain the rest of situations from I_4 on. ∎

The anomalous case of the original formulation of the Yale Shooting Problem would correspond to J, with $J_0 = I_0$, $J_1 = I_1$ and:

$$\begin{aligned} J_2 &= \{Alive, \neg Loaded, Shoot\} \\ J_3 &= \{Alive, \neg Loaded\} \end{aligned}$$

that is, the gun gets unloaded after executing action wait. We can easily see that (J) is not a temporal answer set of T_{YS}, since fixing J as dual world, we can obtain a lower model (J^a, J^a, J) where we do not have any literal for $Loaded$ at J_2^a.

As a final remark, notice that narratives in TNF$_3$ have always an infinite length. However, in many narrative-based action approaches, models are understood as a finite sequence of states. To cover this case, we add a slight variation to TNF$_3$ that allows both finite and infinite narratives by just modifying the interpretation of $\bigcirc$ operator:

$$7)\ \ (\bigcirc\phi)^{x_j} = \begin{cases} 1 & \text{if} j = n \text{ (being n, the last position)} \\ \phi^{x_{j+1}} & \text{otherwise} \end{cases}$$

and assuming that, in condition 8), we require $j \leq k \leq n$ when the narrative has finite length n. If we are in the final situation, then any $\bigcirc$ formula will be true and so, we can define $\mathbf{end} \stackrel{\text{def}}{=} \bigcirc\bot$. Finite or infinite length narratives can be forced by respectively adding the standard axiom $\Diamond\mathbf{end}$ or its negation.

4 An example of use: translation of language $\mathcal{B}$

As an example of how TNF$_3$ can be easily used for translating action description languages, we propose here an encoding of language $\mathcal{B}$ introduced in [15]. Language $\mathcal{B}$ is a simple modification of the well known language $\mathcal{A}$ for treating indirect effects by interpreting causal laws as inference rules (with the same orientation of [21, 2, 29]). It defines a two-sorted signature: the set $\mathbf{A}$ of action names and the set $\mathbf{F}$ of fluent names. An action description $D = Y \cup Z$ consists of a set Y of dynamic laws:

$$A \ \mathbf{causes}\ L \ \mathbf{if}\ F$$

and a set Z of static laws:

$$L \ \mathbf{if}\ F$$

being $A \in \mathbf{A}$, L a fluent literal and F a conjunction of fluent literals. The semantics of $\mathcal{B}$ is defined as follows. A set s of literals is closed under Z iff for each $(L\ \mathbf{if}\ F) \in Z$ such that s satisfies F, $L \in s$. Let $Cn_Z(s)$ be the smallest set of literals containing s that is closed under Z.

For a given action description $D = Y \cup Z$, we define a transition system as a triple $\langle S, V, R\rangle$ where:

1. S is the set of all (2-valued) interpretations for $\mathbf{F}$ that are closed under Z.

2. $V(P, s) = s(P)$ is a valuation function that assigns a truth value ($true$ or $false$), to each fluent P at each state $s \in S$.

3. R is the transition relation containing all the triples $\langle s, A, s'\rangle$ such that:

$$s' = Cn_Z(E(A, s) \cup (s \cap s')) \tag{7}$$

where:

$$E(A, s) = \{L : (A\ \mathbf{causes}\ L\ \mathbf{if}\ F) \in Y \text{ and } s \text{ satisfies } F)\} \tag{8}$$

The translation into TNF_3 is defined as follows. For each static or dynamic law $\phi \in D$ we define $\Pi(\phi)$ as:

$$\Pi(L\ \mathbf{if}\ F) = \Box(L \leftarrow F) \tag{9}$$

$$\Pi(A\ \mathbf{causes}\ L\ \mathbf{if}\ F) = \Box(\bigcirc L \leftarrow A, F) \tag{10}$$

The TNF_3 theory $\Pi(D)$ is defined as $\{\Pi(\phi) : \phi \in D\}$ plus inertia for all fluents (rules (5) and (6)), axiom $\Diamond\mathbf{end}$ and the axiom schemata:

$$\neg P | P \leftarrow \top \tag{11}$$

$$\Box(\neg A | A \leftarrow \top) \tag{12}$$

$$\Box(\bot \leftarrow A_i, A_j) \tag{13}$$

$$\Box(\neg A \leftarrow \mathbf{end}) \tag{14}$$

for each $P \in \mathbf{F}$, and $A, A_i, A_j \in \mathbf{A}$, with $A_i \neq A_j$. Rules (11) and (12) complete the literals for fluents at the initial situation and the literals for action occurrences at any situation respectively. Rule (13) avoids concurrence of actions. Finally, rule (14) implies that no action is executed at the last situation in the narrative.

We will establish a correspondence between transitions and subsequences of narratives so that, given a transition $\langle s, A, s' \rangle$ its corresponding subsequence is (I_j, I_{j+1}), where:

1. $I_j = s \cup \{A\} \cup \{\neg A' : A' \in \mathbf{A}, A' \neq A\}$ (only action A occurs)
2. $s' \subseteq I_{j+1}$

Notice that I_{j+1} is actually a set of states in which we leave free the next action to be executed.

Theorem 1 *Given an action description D in $\mathcal{B}$, transitions $\langle s, A, s' \rangle$ correspond to temporal answer sets of $\Pi(D)$ plus the axiom $\neg\mathbf{end} \wedge \bigcirc\mathbf{end}$, which fixes finite narratives of length 2.*

Proof When the narrative length is 2, we may instantiate $\Pi(D)$ transforming $\Box$ rules into:

$$
\begin{array}{rcll}
\Pi(L\ \mathbf{if}\ F) & = & \{L \leftarrow F, & (15)\\
 & & \bigcirc L \leftarrow \bigcirc F\} & (16)\\
\Pi(A\ \mathbf{causes}\ L\ \mathbf{if}\ F) & = & \bigcirc L \leftarrow A, F & (17)\\
\bigcirc P & \leftarrow & \mathit{not}\ \bigcirc \neg P, P & (18)\\
\bigcirc \neg P & \leftarrow & \mathit{not}\ \bigcirc P, \neg P & (19)\\
\neg A | A & \leftarrow & \top & (20)\\
\bigcirc \neg A | \bigcirc A & \leftarrow & \top & (21)\\
\bot & \leftarrow & A_i, A_j & (22)\\
\bot & \leftarrow & \bigcirc A_i, \bigcirc A_j & (23)\\
\bigcirc \neg A & \leftarrow & \bigcirc \mathbf{end} & (24)
\end{array}
$$

where we have replaced each $\Box\phi$ rule by ϕ and $\bigcirc\phi$ and have omitted some resulting rules because of being trivially true for narratives of length 2.

Let (I), $I = \{I_0, I_1\}$, be a temporal answer set of $\Pi(D)$. Let s and s' be the fluent literals contained in I_0 and I_1 respectively. It can be easily seen that rule (11) guarantees that s is complete for the set of fluents $\mathbf{F}$. Besides, given rules (15), s is closed with respect to Z (this is due to minimality of a TNF$_3$ selected model and the interpretation of $\leftarrow$). So, all the possible combinations of fluent literals in I_0 are all the $s \in S$ closed under Z.

Now, notice that rules (20) and (22) guarantee that in I_0 some unique action is true, whereas rule (24) implies that no action is true in I_1. This means that all the possible I_0 correspond to all the possible pairs $\langle s, A\rangle$, being $s \in S$ closed under Z, whereas in I_1 we can just take into account the fluent literals s'.

Finally, for any literal $L \in E(A, s)$ we will have that its corresponding rule of shape (16) will have its condition true in I_0, and so, we have $(\bigcirc L \leftarrow \top)$. Let L be a fluent literal in $(s \cap s')$. If L is an atom P, assuming s' not to be inconsistent, $\neg P \notin s'$, and condition of rule (18) will be true leading to $(\bigcirc P \leftarrow \top)$. A similar reasoning can be made for $L = \neg P$. When s' is inconsistent, then both s and s' are the set of all literals, and so, it is clear that $\bigcirc L \leftarrow \top$ for all $L \in (s \cap s')$. Thus, for a given $\langle s, A, s'\rangle$ we have a program with $\bigcirc L \leftarrow \top$ for any $L \in E(A, s) \cup (s \cap s')$. Since static laws Z are also translated for situation I_1 (rules (16)), it follows that $s' = Cn_Z(E(A, s) \cup (s \cap s'))$. ■

Together with action description languages, several *query* languages have been proposed [15] relying on the concept of *histories*:

Definition 5 *A* history *of a transition system $\langle S, V, R\rangle$ of length n is a sequence $s_0, A_1, s_1, \ldots, A_n, s_n$, where $(s_0, \ldots, s_n) \in S$, $(A_1, \ldots, A_n) \in \mathbf{A}$ and $n \geq 0$, that satisfies: $\langle s_{i-1}, A_i, s_i\rangle \in R$ for all $(1 \geq i \geq n)$.*

We will study the simplest case of query language, called $\mathcal{Q}$, which is thought for checking actual actions in a narrative and consists of propositional expressions for atoms of the shape:

$$A \textbf{ occurs at } t_i$$
$$P \textbf{ holds at } t_i$$

with $A \in \mathbf{A}$, $P \in \mathbf{F}$ and t_i a non-negative integer ranging from 0 to some fixed length n. A history $s_0, A_1, s_1, \ldots, A_n, s_n$ for a given transition system *satisfies* a query ϕ iff:

1. ϕ has the shape A **occurs at** t_i and $A = A_{i+1}$
2. ϕ has the shape P **holds at** t_i and s_i classically satisfies P
3. if ϕ is non-atomic, it follows the truth tables of propositional logic.

The translation of $\mathcal{Q}$ into TNF$_3$ is quite obvious:

$$\Pi(A \textbf{ occurs at } t_i) = \bigcirc^i A$$
$$\Pi(P \textbf{ holds at } t_i) = \bigcirc^i P$$

Theorem 2 *Given an action description D in $\mathcal{B}$, any temporal answer set of* $\Pi(D)$ *corresponds to a history of D.*

Proof The proof follows by induction on the length of the finite narrative n, taking as the base case theorem 1 and generalizing the program instantiation $\Pi(D)^*$ for any $i > 2$. ■

Theorem 3 *Given an action description D in $\mathcal{B}$, and a query ϕ in $\mathcal{Q}$, any temporal answer set of* $\Pi = \Pi(D) \cup \Pi(\phi)$ *corresponds to a history of D that satisfies ϕ.*

Proof Due to theorem 2, and that $\Pi(\phi)$ does not contain default negation, its addition to $\Pi(D)$ will lead to answer sets which correspond to a subset of the histories of D. Then, the correspondence between histories satisfying ϕ and models of $\Pi(\phi)$ follows from TNF_3 satisfaction of formulas (notice that an action A_i is actually located at state I_{i-1}). ■

5 Relation to TEMPLOG

In this section, we will show that TEMPLOG is subsumed by TNF_3, that is, the latter can be used as a generalization of the former for syntax-independent disjunctive programs with classical and default negation.

The syntax of TEMPLOG is defined as follows. Let P denote an atom and N a next-atom, that is, a formula of the form $\bigcirc^i P$ for some $i > 0$. Let ϵ denote the empty expression. Then, a TEMPLOG program consists of a set of initial, permanent and goal clauses grammatically defined as:

Non-empty Body:	B	$::= P \parallel B_1, B_2 \parallel \bigcirc B \parallel \Diamond B$
Body:	D	$::= \epsilon \parallel B$
Initial clause:	IC	$::= N \leftarrow D \parallel \Box N \leftarrow D$
Permanent clause:	PC	$::= \Box(N \leftarrow D)$
Goal clause:	G	$::= \leftarrow D$

In [3] the *temporal least model* of a TEMPLOG program Π is defined as a natural generalization of the least model of a definite program [9]. The valuation of a formula in TEMPLOG is defined as the usual (2-valued) Propositional Temporal Logic [1]. With respect to TNF_3, it simply corresponds to a restriction of definition 3, disabling the use of $\boldsymbol{d}$ operator and requiring $x = I^a$ to be a complete and consistent set of literals. In this way, for describing TEMPLOG semantics, the usual representation for an interpretation is a sequence of sets of atoms, rather than sets of literals. Due to the properties:

$$\Diamond\phi \quad \Leftrightarrow \quad \bigvee_{i=0}^{\infty} \bigcirc^i \phi \tag{25}$$

$$\Box\phi \quad \Leftrightarrow \quad \bigwedge_{i=0}^{\infty} \bigcirc^i \phi \tag{26}$$

$$\bigcirc(\phi \wedge \psi) \quad \Leftrightarrow \quad \bigcirc\phi \wedge \bigcirc\psi \tag{27}$$

any TEMPLOG program can be equivalently transformed into a (possibly infinite) definite program[2] Π^*, containing a set of clauses in which the only temporal operator occurs in the shape of next-atoms $\bigcirc^i P$. This allows us seeing any next-atom "$\bigcirc^i P$" as a new renamed classical atom, and considering an infinite signature of the shape $\{P, Q, \ldots, \bigcirc P, \bigcirc Q, \ldots, \bigcirc^2 P, \bigcirc^2 Q, \ldots\}$. Under this point of view, Π^* becomes a *definite* logic program (negation does not occur in next-atoms), and so, it is possible to define the least model of Π^*, as a classical program.

Definition 6 **(Temporal Least Model)** *The* temporal least model *of a TEMPLOG program* Π *is defined as the corresponding least model of* Π^* *as a renamed classical program.*

The following result allows the sound transformation of Π into Π^* under the TNF_3 semantics.

Theorem 4 *If* Π *is a conjunction of clauses following the TEMPLOG syntax, then* $\Pi \Leftrightarrow \Pi^*$ *when dealing with infinite narratives.*

Proof It trivially follows from the fact that properties (25), (26) and (27) also hold in TNF_3. ■

Given a set of atoms J, let $\neg\overline{J}$ denote the set $\{\neg P : P \notin J\}$. If I is a set of next-literals, we define $temp(I)$ so that $L \in temp(I)_i$ iff $\bigcirc^i L \in I$.

Theorem 5 *The set of atoms* J *is the temporal least model of a TEMPLOG program* Π *iff the set of literals* $I = J \cup \neg\overline{J}$ *corresponds to the unique* TNF_3 *temporal answer set* $(temp(I))$ *of* $\Pi \cup \{\Box\neg\mathbf{end}\} \cup \{\neg \bigcirc^i P \leftarrow \mathit{not}\ \bigcirc^i P,$ *for any atom* P *and* $i \geq 0\}$.

Proof We will use the transformation into a definite program Π^* and the close relation between stable models and answer sets. First, notice that, since Π^* is definite, its least model coincides with its unique stable model. Second, as discussed in [13] (section 6) a stable model J of any program without classical negation corresponds to the answer set $J \cup \neg\overline{J}$ of that same program after adding the closed world assumption, $\neg P \leftarrow \mathit{not}\, P$, for any atom P. ■

6 Discussion and related work

Several interesting comparisons may be outlined depending on two different points of view: the static formulation of answer sets and the temporal extension. In the static case, the introduced approach NF_3 is clearly inspired to the already mentioned work of MBNF [18], a bi-modal logic relying on two modal operators, B and *not* , which respectively stand for belief and default negation. As already shown in [18], DELP can be formulated as MBNF *theories with protected literals*, that is, propositional combinations of literals in the scope of one modal operator. From this result, given a

[2] See [3] for further details.

literal L, we can directly establish the relation between BL (resp. $not\, L$) in MBNF and $\boldsymbol{l}L$ (resp. $not\ L$) in NF_3. In fact, this correspondence is also applicable for nested modal operators. However, when we have a general propositional formula like for instance $\phi = P \vee \neg Q$, then $\boldsymbol{l}\phi$ is equivalent to $\boldsymbol{l}P \vee \boldsymbol{l}\neg Q$ which clearly does not correspond to $B\phi$ any more. This is due to the fact that NF_3 relies on three-valued logic, while MBNF handles S5-like modal frames. Thus, NF_3 can be seen as a portion of MBNF dedicated to reasoning about literals. There exist multiple three-valued reformulations of answer sets (see for instance, [4]). Among them, the most closely related to NF_3 is the so called Equilibrium Logic presented in [26]. Equilibrium Logic is based on a strong negation variation of Intuitionistic Logic which handles a chain of just two worlds (called "here" and "there") corresponding to our actual and dual world. With respect to usual three-valued logic, it introduces two new operators not_e and $\rightarrow_e$ (in fact, $not_e\ \phi$ can be defined as $\phi \rightarrow_e \bot$). In its use for logic programming, we can establish the following relation wrt NF_3:

i) $not_e\ \phi$ corresponds to $\neg \boldsymbol{l}\phi \wedge \boldsymbol{d}\neg \boldsymbol{l}\phi$, that is, $\neg \boldsymbol{l}\phi \wedge not\ \phi$

ii) $\phi \rightarrow \psi$ corresponds to $(\psi \leftarrow \phi) \wedge \boldsymbol{d}(\psi \leftarrow \phi)$

As we can see, Equilibrium formulas are a strengthening of NF_3 where we require each negation to be true not only in the dual world but also in the actual one, and, in a similar way, we require each rule to be true not only the actual world, but also in the dual one. Though this reinforcement is not present in the answer sets definition, the obtained results are similar. In this sense, NF_3 is closer to the original answer sets definition and also to its MBNF encoding.

As for the relation to other temporal approaches, though we have already proved that TNF_3 is an extension of TEMPLOG, there exist many other logic programming frameworks for temporal logic [10, 24, 11, 25] or even general modal logic [7] though none of them combine the simplicity of linear-time modal logic with the availability of a declarative formulation for default negation.

Inside the area of application to Reasoning about Actions, the most similar approach to the one presented here is, perhaps, the Past Temporal Logic for Actions [23] (PTLA). It is also based on linear time temporal logic, though it uses the past operators instead of the future ones used in TNF_3. In fact, an extension of TNF_3 for past operators is quite direct, as already was proposed for TEMPLOG. The main difference of TNF_3 with respect to PTLA is that, in the latter, the surface action language is modal, while its logic programming translation uses timepoints in order to represent situations.

Another great group of approaches for Reasoning about Actions are those related to Situation Calculus. The main difference with respect to the work presented here is that they consider actions as a special kind of objects intimately related to the representation of situations. Our orientation has been, on the contrary, closer to narrative-based formulations, in which actions occur in a situation in a similar way to fluents. This simplifies representational issues like concurrence of actions or existence of possible situations. Inside the Situation Calculus trend, we can distinguish between nonmodal approaches, which follow the original Situation Calculus formulation, and modal approaches, which are based on variations of Harel's Dynamic logic.

The most representative case of the nonmodal approaches is GOLOG [17], a logic programming language based on a variation of Situation Calculus, for which an interpreter programmed in Prolog has been constructed. This approach, however, has a completely different orientation to the work presented here, since the semantics of GOLOG is not actually based on usual logic programming semantics, but in second order logic instead. In fact, GOLOG does not directly deal with typical logic programming issues like default negation semantics. Another nonmodal approach which is perhaps more in the direction of this work is [29], where Situation Calculus theories are represented using extended logic programs under the answer sets semantics. As for approaches for Reasoning about Actions based on Dynamic Logic we could cite, for instance, [20, 27, 6], though only the first one relies on a logic programming paradigm.

Future work will be focussed on two directions. On the one hand, comparisons to other action approaches will be analyzed, either for narrative ones like the Event Calculus [28] or for non-narrative approaches, which will require an extension of TNF_3 to deal with hypothetical reasoning (as introduced for instance in [2]). On the other hand, we will deepen on implementation issues, trying to extend the temporal resolution query procedure of TEMPLOG, temporal SLD, for dealing with default negation.

Acknowledgements I wish to thank Ramon P. Otero, David Lorenzo and Ale Provetti for their technical assistance and presentation review. Thanks also to Vladimir Lifschitz and the AAAI for the travel support for attending NMR-98 workshop at Trento. This research is partially supported by the Government of Spain, grant PB97-0228.

References

[1] M. Abadi and Z. Manna. Temporal logic programming. *Journal of Symbolic Computation*, 8:277–295, 1989.

[2] C. Baral, M. Gelfond, and A. Provetti. Reasoning about actions: Laws, observations and hypotheses. *Journal of Logic Programming*, 31, 1997.

[3] M. Baudinet. A simple proof of the completeness of temporal logic programming. In L. Fariñas del Cerro and M. Penttonen, editors, *Intensional Logics for Programming*, pages 51–83. Clarendon Press, Oxford, 1992.

[4] P. A. Bonatti and L. Giordano. Three-valued semantics for extended logic programming. In *Proc. joint Conf. on Declarative Programming GULP-PRODE'95*, 1995.

[5] R. Bull and K. Segerberg. Basic modal logic. In D.M. Gabbay and F. Guenthner, editors, *Handbook of Philosophical Logic*, volume 2, pages 1–88. D. Reidel Publishing Company, 1984.

[6] Marcos A. Castilho, Olivier Gasquet, and Andreas Herzig. Formalizing action and change in modal logic I: the frame problem. *Journal of Logic and Computation*, to appear.

[7] L. Fariñas del Cerro. MOLOG: A system that extends prolog with modal logic. *New Generation Computing*, 4:35–50, 1986.

[8] J. Dix, L. Pereira, and T. Przymusinski. Prolegomena to logic programming for non-monotonic reasoning. In In J. Dix, L. Pereira, and T. Przymusinski, editors, *Nonmonotonic Extensions of Logic Programming, LNAI 1216*, pages 1–36. Springer-Verlag, 1997.

[9] M. H. Van Emden and R. A. Kowalski. The semantics of predicate logic as a programming language. *Journal of ACM*, 23(4):733–742, October 1976.

[10] M. Fujita, S. Kono, T. Tanaka, and T. Moto-oka. Tokio: Logic programming language based on temporal logic and its compilation into prolog. In *Proc. of the 3rd Intl. Conf. on Logic Programming (LNAI 225)*. Springer-Verlag, 1986.

[11] D. Gabbay. Modal and temporal logic programming. In A. Galton, editor, *Temporal Logics and their Applications*, chapter 6, pages 197–237. Academic Press, 1987.

[12] M. Gelfond and V. Lifschitz. The stable model semantics for logic programming. In R. A. Kowalski and K. A. Bowen, editors, *Logic Programming: Proc. of the Fifth International Conference and Symposium (Volume 2)*, pages 1070–1080. MIT Press, Cambridge, MA, 1988.

[13] M. Gelfond and V. Lifschitz. Classical negation in logic programs and disjunctive databases. *New Generation Computing*, 9:365–385, 1991.

[14] M. Gelfond and V. Lifschitz. Representing action and change by logic programs. *The Journal of Logic Programming*, 17:301–321, 1993.

[15] M. Gelfond and V. Lifschitz. Action languages. *Linköping Electronic Articles in Computer and Information Science*, 3(16), October 1998. `http://www.ep.lius.se/ea/cis/1998/016/`.

[16] S. Hanks and D. McDermott. Nonmonotonic logic and temporal projection. *Artificial Intelligence*, 33:379–412, 1987.

[17] H. Levesque, F. Pirri, and R. Reiter. Foundations for the situation calculus. *Linköping Electronic Articles in Computer and Information Science*, 3(18), December 1998. `http://www.ep.lius.se/ea/cis/1998/018/`.

[18] V. Lifschitz. Minimal belief and negation as failure. *Artificial Intelligence*, 70:53–72, 1994.

[19] V. Lifschitz. *Action languages, answer sets and planning.* Springer Verlag, 1999.

[20] Alberto Martelli Matteo Baldoni, Laura Giordano and Viviana Patti. An abductive proof procedure for reasoning about actions in modal logic programming. In *Proc. of Non-Monotonic Extensions on Logic Programming (NMELP'96)*, 1996.

[21] Norman McCain and Hudson Turner. A causal theory of ramifications and qualifications. In C. S. Mellish, editor, *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI)*, pages 1978–1984, Montreal, Canada, August 1995. Morgan Kaufmann.

[22] John McCarthy and Patrick J. Hayes. Some philosophical problems from the standpoint of artificial intelligence. In B. Meltzer and D. Michie, editors, *Machine Intelligence 4*, pages 463–502. Edinburgh University Press, 1969.

[23] G. Mendez, J. Llopis, J. Lobo, and C. Baral. Temporal logic and reasoning about actions. In *Proc. of Common Sense 96*, 1996.

[24] B. Moszkowski. *Executing Temporal Logic Programs.* Cambridge University Press, 1986.

[25] M. A. Orgun and W. W. Wadge. Theory and practice of temporal logic programming. In L. Fariñas del Cerro and M. Penttonen, editors, *Intensional Logics for Programming*, pages 21–50. Clarendon Press, Oxford, 1992.

[26] D. Pearce. A new logical characterisation of stable models and answer sets. In *Non monotonic extensions of logic programming. Proc. NMELP'96. (LNAI 1216).* Springer-Verlag, 1997.

[27] C. B. Schwind. A logic based framework for action theories. In J. Ginzburg, Z. Khasidashvili, C. Vogel, J.-J. Lévy, and E. Vallduví, editors, *The Tbilisi Symposium on Logic, Language and Computation*, pages 275–291. 1997.

[28] M. Shanahan. *Solving the Frame Problem.* MIT press, 1997.

[29] H. Turner. Representing actions in logic programs and default theories: A situation calculus approach. *Journal of Logic Programming*, 31:245–298, 1997.

[30] A. van Gelder, K. A. Ross, and J. S. Schlipf. The well-founded semantics for general logic programs. *Journal of the ACM*, 38(3):620–650, 1991.