Annals of Mathematics and Artificial Intelligence manuscript No.
(will be inserted by the editor)

t-DeLP: an argumentation-based Temporal Defeasible Logic Programming framework

Pere Pardo · Lluís Godo

Received: date / Accepted: date

Abstract The aim of this paper is to propose an argumentation-based defeasible logic, called t-DeLP, that focuses on forward temporal reasoning for causal inference. We extend the language of the DeLP logical framework by associating temporal parameters to literals. A temporal logic program is a set of basic temporal facts and (strict or defeasible) durative rules. Facts and rules combine into durative arguments representing temporal processes. As usual, a dialectical procedure determines which arguments are undefeated, and hence which literals are warranted, or defeasibly follow from the program. t-DeLP, though, slightly differs from DeLP in order to accommodate temporal aspects, like the persistence of facts. The output of a t-DeLP program is a set of warranted literals, which is first shown to be non-contradictory and be closed under sub-arguments. This basic framework is then modified to deal with programs whose strict rules encode mutex constraints. The resulting framework is shown to satisfy stronger logical properties like indirect consistency and closure.

Keywords Logic programming · Argumentation · Non-monotonic logic · Temporal Reasoning

Mathematics Subject Classification (2000) 03B44 · 03B53

This paper is a revised and extended version of the conference paper [41] which was presented at the 5th International Conference on Scalable Uncertainty Management (SUM 2011).

P. Pardo
Institut d'Investigació en Intel·ligència Artificial (IIIA - CSIC), E-08193 Bellaterra, Spain
and Dept. de Lògica, Hist. i Filo. Ciència, Universitat de Barcelona, E-08001 Barcelona, Spain
E-mail: pardo@iiia.csic.es

L. Godo
Institut d'Investigació en Intel·ligència Artificial (IIIA - CSIC), E-08193 Bellaterra, Spain
E-mail: godo@iiia.csic.es

1 Introduction

In this contribution, we present an argumentation-based temporal defeasible logic, called t-DeLP, mainly inspired by DeLP [23] but with a focus on causal temporal reasoning. The language is defined by a set of temporal literals -representing facts-, and (strict or defeasible) durative rules. A temporal logic program consists of temporal facts and rules, which combine into arguments for further derivable facts. The main motivation for t-DeLP is to reason about interacting processes (modeled as arguments), and then decide which arguments (conclusions) are to prevail. An argument expresses some delay between each premise (cause) and the conclusion (effect), thus suggesting how a process might evolve. Since different arguments (process descriptions) might conflict, a dialectical procedure is proposed that decides which arguments prevail based on their conflicts. The conclusions of undefeated arguments define the set of warranted effects (defeasible logical consequences) of a t-DeLP program. These consequences thus describe the actual states that will turn up, according to the available information about the initial state(s) and the temporal rules.

A motivation for *defeasible* logics (like for the family of non-monotonic logics) is the descriptive parsimony it allows for knowledge bases. This parsimony is in accordance with everyday causal reasoning, where it is standard practice to list only those causes that are uncommon or just specific to the process: e.g. a *spark* caused a *fire*. Causes that usually hold, like *oxygen*, are not mentioned in the explanation (or rules) unless they are false and this explains the non-occurrence of the effect: the *spark* did not start a *fire* because *no-oxygen*. Thus, we aim to describe a general-purpose logic to reason about temporal or causal processes, leaving to the user the particular level-of-detail of explanations. Besides these questions of what is to occur afterwards, e.g. whether *fire* or *no-fire*, another motivation exists for a specialized study of temporal aspects within this non-monotonic stance. Namely, the question whether *fire* will start at some future time t or instead at $t + 1$ or $t + 2$, etc. This is important in scenarios where the start and duration of a process depends on the initial context (e.g *combustible, fire-retardant*), or on the existence of other processes that are also occurring in parallel (*rain, wind*).

A well-known contribution among argumentation-based defeasible logics is that of García and Simari's [23]. The authors present DeLP, a logic programming formalism based on defeasible argumentation. The question of how to define the defeat or preference relation between arguments is also discussed at length in this work. Inspired by Poole [42], the authors of [49] focus on a formal criteria called *generalized specificity*, which gives preference to arguments with more premises or more direct rules. [1] But the latter seems at odds with causal reasoning in a temporal setting: we would rather prefer *less direct* rules, i.e. more detailed temporal inferences. [2] Thus, we adapt this and other aspects of DeLP to the temporal case in order to intuitively meet basic intuitions about causal explanations. Some of these differences arise from

[1] This criterion captures the preference for e.g. {*penguins do not fly*} over {*penguins are birds*, *birds fly*} in evidence-based reasoning, not considered here.

[2] More direct rules can fail to detect interactions. Consider, for instance, two moving objects that are directed against each other. Under non-detailed rules, these objects would *magically* not collide but reach their destinations untroubled.

the temporal asymmetry (past vs. future) of causation: persistence, the attack relation and defeat deserve special attention for the temporal case. As a consequence, the notion of warrant for (temporal) literals is slightly different from that of DeLP as presented in [23].

1.1 Structure of the paper

The paper is structured as follows. After Section 2 on related work, some preliminaries on notation and knowledge representation are presented in Section 3. Then, we present in Section 4 the t-DeLP logic programming framework. In Section 5 we show the basic argumentation-theoretic properties of warrant for t-DeLP programs: direct consistency and closure under sub-arguments. Finally, in Section 6, we present a modification of the t-DeLP algorithm for warrant, in order to deal with programs whose strict rules encode binary mutex constraints. For this extended framework, we prove that the notion of warrant satisfies direct consistency, indirect consistency and closure. The paper ends with Section 7, where t-DeLP is compared with Dung semantics [21], with DeLP regarding the defeat criteria (other elements being compared throughout the paper) and finally with the closely related logic programming framework TDR [5]. After the Conclusions section, we add an Appendix containing the proofs of the auxiliary results needed in this paper.

2 Related Work

There is a vast literature on logics for reasoning about events, roughly dividing into three areas: temporal logics, causal or conditional logics, and logics for actions. The former two put the focus on the temporal aspects of change, and resp. the causal relationships between state conditions. This is sometimes done by focusing on states and leaving events (transitions between states) without an explicit representation in the object language, as in the planning tradition. Logics for actions, on the other hand, aim to capture the effects of complex actions by making action events explicit. In comparison to temporal approaches, this is usually done under quite abstract notions of time (*before vs. after this action*).

Modal logic [15], [8] is one of the most central areas within logic in computer science, and has been used in particular for the study of actions and events (among many other topics). For example, studies in applied modal logic include modalities for time in linear time LTL [29], or branching time CTL, CTL* [22]; or modalities for the execution of programs PDL [31]. The rather comprehensive knowledge on the area of modal logic has turned an advantage to the corresponding studies in these and many other topics.

In practice, though, the early discovery of some knowledge engineering problems (conditionals, incomplete knowledge, the frame problem) motivated a switch from (classical) implication-based logics to non-monotonic rule-based systems (e.g. default logic [40]). Let us briefly recall these problems:

- *conditionals* cannot be faithfully modeled in monotonic (modal) logic due to the failure of antecedent strengthening (e.g. *if I became rich, you would be happy*; *if I became rich and greedy, you would not be happy*.)
- *incomplete knowledge*, i.e. a substantial lack of knowledge, does not prevent commonsense reasoners to extract (possibly false) information by way of (possibly unsound) inferences, e.g. using default rules. These maneuvers also make sense for economic reasons: we can eliminate conditions for an event from its description, if these are usually known to hold; and so we can forget about checking these default conditions.

The first works addressing these problems are rather old, see e.g. [36] and [47]. Both problems are about the existence of some sort of priority among the set of contingently false states and, resp., unknown facts. Thus, while standard approaches to conditionals are based on a preference for more similar worlds (among those satisfying the antecedent), default logics assign more plausibility to default rules or normal facts than to their contraries (in case of ignorance).

Another motivation for the study of non-monotonic reasoning was the recognition of the *frame problem*. In the broad sense, this denotes a family of problems related to the description of actions: their effects, non-effects, or preconditions. Among them, we find:

- the *frame problem* -in the original, narrow sense- is the problem of finding (efficient) representations for the persistence of facts through time or action executions. Since an action will only change a small part of a scenario, it is unpractical to make an explicit list of which facts persist under which actions, to be used during inference.
- the *ramification problem* is the problem of efficiently deriving the indirect effects of an action from a list of direct effects (e.g. possibly using laws). Otherwise, one must explicitly list all the effects of an action as direct effects.
- the *qualification problem* is that of finding efficient representations for the preconditions of an action. We would also like to prune many of the preconditions that one would not bother to check before the action (unless one positively knows about their failure). Among these three problems, this is the only one apparently requiring a non-monotonic approach [14].

The original frame problem plagued the initial classical logic based approaches [38], etc. and also many (monotonic) temporal logic programming works [1]. Despite considerable efforts were made to solve the frame problem, solutions were not completely satisfactory w.r.t. all the variations of this problem. More recently, some efforts have been devoted to solve the frame problem within monotonic modal logics. For example, some of the issues related to the frame problem in PDL have been successfully dealt with in [45], [26], [51], [14]. Other research areas, instead, have found natural ways to avoid the frame problem (as in the area of automated planning [25]) or address it in natural ways. Among the latter we find non-monotonic logics (see below). These logics have found more natural expressions to capture common sense reasoning and avoid knowledge representation issues.

The present work belongs to the area of non-monotonic temporal logics, where non-monotonicity here is built upon the recent area of computational argumentation

[21], [43], [46] and more specifically under the form of logic programming [12], [23]. In t-DeLP we do not represent events or actions explicitly, so a t-DeLP notion of update (computing the results of executing a program) would rather be similar to that of planning: a state-transition system, with some incorporated notion of t-DeLP consequence. Though this topic falls out of the scope of the paper, we present an example for this, based on the Yale shooting scenario. On the other hand, we hope to make clear that the above representation problems can be addressed when actions are considered: the (narrow) frame problem can be solved by means of persistence rules; the issue of conditionals and the ramification problem can be addressed by introducing rules that encode appropriate laws; and the qualification problem can be solved by the argumentation procedure, since arguments about preconditions can qualify whether an action is executable.

Let us then briefly survey different non-monotonic logics frameworks, and more specifically those addressing reasoning about events, action or time. Non-monotonic reasoning (see [10]) became, for the reasons exposed, another important area of research on the present topics. Inspired by common sense reasoning, these logics are based on the existence of priorities between inferences [44], [3], [39]. Thus, while all inferences separately make sense, some of them might be preferred to (and cancel) others. This is useful in case a logical conflict exists between inferences; e.g. *typically birds fly*, and *penguins are birds* are in conflict with *penguins do not fly*. In this line, for example, one can find early approaches to counterfactual reasoning [36], and default reasoning [47]. Non-monotonic logics, though, have evolved into a rather disperse variety of logical approaches, including some modal approaches [34], [27]. This makes it difficult to reduce non-monotonic systems to each other or to modal logics, though some correspondences or reductions are known. For example: default logics into autoepistemic logics [34] or into DeLP [20], DeLP into answer set programming ASP [50], the relation between DeLP and normal logic programing [16], defeasible logic and definite logic programming [2]. Among non-monotonic modal approaches, we find [27],[11]. Other logics of action and causation include $\mathcal{C}/\mathcal{C}+$ [28], [19], $\mathcal{A}$ [24], event calculus [35] and others. These have been studied from the standpoint of PDL in [51].

Further motivations for the present, argumentation-based approach are precisely questions on these priorities between conflicting inferences [48], [23]: how are they defined, but also how can they be automatically generated, etc. In most approaches, answers to the second question have been traditionally assumed as given. In particular, this is the case of abstract argumentation frameworks [21], and its extension with preferences [4]. These questions are certainly pertinent when one does consider the internal structure of inferences: in default logics [47] or in logic-based argumentation [12], [43]. For example, this internal structure of arguments would permit to decompose the preference between *arguments* into an aggregation of preferences between *their components*, as proposed in [23].

Yet another approach for priorities consists in comparing the information contained in two arguments to decide between these two. Inspired by the notion of *specificity* [42], Chesñevar et al. [49] explored the idea of a formal, general-purpose preference relation for priorities. This led to the defeasible argumentation-based DeLP logic programming framework García and Simari [23]. The proposed framework

t-DeLP consists, roughly, in extending the language of DeLP with temporal information, and adaptating the preference relation to causal temporal reasoning. This contrasts with a focus on evidence-based reasoning, which traditionally motivated many non-monotonic logical systems. For a comparison between DeLP and t-DeLP in this respect, see Section 7.

Within the area of defeasible logics, rule-based systems were initially proposed (see Billington [7] and Nute [40]). These were recently extended to reason with temporal literals in Governatori and Terenziani [30]. Indeed, our language is mainly inspired by this latter work. Rule-based systems, though, lack the simplicity and power of argumentation-based logical frameworks. First, rule-based systems are based on rules and defeaters, resp. to promote and prevent derivations. In contrast, argumentative frameworks only consist of rules, in a way that mirrors a deliberating human agent pondering reasons for and against candidate conclusions. Also, argumentation-based logics are more powerful than rule-based systems: while priority relations between rules establish local comparisons or preferences, preference relations between arguments can be defined globally. Thus, arguments can be judged according to how much information they make use of, even if at the level of rules, they cannot be so compared. The latter question applies to other frameworks for reasoning with temporal information, either defeasible or based on belief change operators, e.g. Hunter [32], [33].

Finally, several frameworks have been proposed in the more recent area of logical models of argumentation. The initial approach of Dung [21] is based on abstract relations of attack between arguments. This has been extended with temporal parameters describing for which time intervals an argument is available or can be legitimately stated, Cobo et al. [18], [17]. As we mentioned, Dung's work [21] was also extended in the sense of considering arguments as structured by logical elements, e.g. Caminada and Amgoud [12], and Prakken [43] (or Besnard and Hunter [6] for classical logic). This internal logical structure explains both the relation of attack (logical conflict), and the comparison-based relation of preference (that would turn an attack into a defeat). Several works in this area address temporal argumentation as well, but most of them do so by associating time intervals to literals and arguments, see for instance Augusto and Simari [5], Mann and Hunter [37]. Our approach differs from these works in that the *interval where the conclusion of an argument holds*, rather than being a primitive notion, obtains from different arguments (one for each time-point). In this sense, while these works are slightly more expressive than the present proposal, t-DeLP can be seen in some respects as a more faithful (and simpler) formalization of non-monotonic reasoning about temporal events. Concerning expressivity, our discrete time-point based approach in fact accommodates most of the interesting features from temporal rule-based systems and interval-valued argumentation frameworks [30], [5] (expiring literals, persistence).

3 Knowledge Representation

Concerning notation, throughout the paper we make use of the following conventions: strong negation is denoted $\sim p$, for a propositional variable $p \in \mathsf{Var}$. Given two sets

X, Y we denote the set-theoretic difference as $X \setminus Y$ and the Cartesian product of X and Y as $X \times Y$. Sequences are denoted $\langle x_0, \ldots, x_n \rangle$ or $[x_0, \ldots, x_n]$. Given a sequence $\mathbf{x} = \langle x_0, \ldots, x_n \rangle$ and an element x, we denote by $\mathbf{x}^\frown\langle x \rangle$ the concatenation of $\mathbf{x}$ with x, i.e. the sequence $\langle x_0, \ldots, x_n, x \rangle$ or $[x_0, \ldots, x_n, x]$. If f is a function $f : X \to Y$ and $X' \subseteq X$, we define $f[X'] = \{f(a) \in Y \mid a \in X'\}$. Given a family of sets $\mathbf{M}$, its union is denoted $\bigcup \mathbf{M}$.

After fixing the symbols used in this paper, let us describe with more detail the language of our object of study t-DeLP and some representational issues related to the argumentation-theoretic properties [12].

Our language builds upon a set of temporal literals, consisting of a pair $\langle$*literal, time*$\rangle$. Literals are expressions of the form p or $\sim p$ from a given set of variables $p \in \mathsf{Var}$. Strong negation $\sim$ cannot be nested, so we will use the following notation over literals: if $\ell = p$ then $\sim\ell$ will denote $\sim p$, and if $\ell = \sim p$ then $\sim\ell$ will denote p. These literals, though, might rather be seen as ground predicates, of the form *literal* = (*object, property*) or also *literal* = (*object, parameter, value*).

Temporal parameters will take discrete values and will be denoted with t (possibly with subindexes). Thus, a temporal literal is of the form $\langle \ell, t \rangle$. Time is relevant to determine whether a pair of temporal literals contradict each other: for this contradiction to exist, the literals expressed must be the negation of each other *and* they must be claimed to hold at the same time: $\langle \ell, t \rangle$ and $\langle \sim\ell, t \rangle$ are contradictory. A temporal or causal statement (possibly an instance of some general law) is represented as a rule: *a set of temporal literals* $\langle \ell_1, t_1 \rangle, \ldots, \langle \ell_n, t_n \rangle$ *imply a temporal literal* $\langle \ell, t \rangle$. In particular, rules with no duration, i.e. with no delay between their head and body, describe static constraints. Namely, rules $\langle \ell, t \rangle \prec\!\!\!- \langle \ell_1, t_1 \rangle, \ldots, \langle \ell_n, t_n \rangle$, or with $\leftarrow$, such that $t = t_1 = \cdots = t_n$ describe static defeasible or strict constraints within this t.

In any case, literals of the form $\langle$(*object, parameter, value*)*, time*$\rangle$ tacitly require some "logical" constraints to be satisfied: an object cannot have different values of a given parameter at a given time (or, in some cases, two objects cannot have the same value; e.g. for spatial location). These absolute constraints, represented by strict rules, can also be seen as induced by a family of sets of *pairwise* incompatible literals $X = \{(o, p, v), (o, p, v'), \ldots\}$, for fixed p and o (or for fixed p and v); these literals are also called *mutex* in the literature, for <u>mut</u>ual <u>ex</u>clusion.

Example 1 Let $\mathcal{O}$ and $\mathcal{L}$ be the sets of objects o and locations l; and let $@(o, l) \in \mathsf{Var}$ denote: *o is at l*;

- the *at most one location per object* policy is defined by a set $\{o\} \times \mathcal{L}$ for each $o \in \mathcal{O}$; this set corresponds to the set of rules $\langle \sim @(o, l), t \rangle \leftarrow \langle @(o, l'), t \rangle$, for each $l \neq l'$.
- the *at most one object per location* policy is defined by a set $\mathcal{O} \times \{l\}$ for each $l \in \mathcal{L}$; this set corresponds to the set of rules $\langle \sim @(o, l), t \rangle \leftarrow \langle @(o', l), t \rangle$, for each $o \neq o'$.

Specific results for t-DeLP programs containing such static strict rules are addressed later in Section 6, though this will be actually done under the form of mutex constraints, rather than as static strict rules like in Example 1.

4 t-DeLP: defeasible logic with (discrete) time.

In a sketch, argumentation-based logic programming formalisms work as follows: we start with a knowledge base (Π, Δ) with temporal facts and rules, and a query $\langle \ell, t \rangle$; we combine facts and rules in (Π, Δ) into an argument $\mathcal{A}$, i.e. a set $\mathcal{A} \subseteq \Pi \cup \Delta$ that entails the (presumable) fact $\langle \ell, t \rangle$ by applying *modus ponens* from $\mathcal{A}$. Once some such argument $\mathcal{A}$ for $\langle \ell, t \rangle$ is fixed, an argumentative process generates counter-arguments $\mathcal{B} \subseteq \Pi \cup \Delta$ defeating $\mathcal{A}$; that is, with $\mathcal{B}$ concluding the contrary of $\langle \ell, t \rangle$ or of some intermediate step in $\mathcal{A}$. Then arguments $\mathcal{C}$ defending $\mathcal{A}$ by way of attacking some such $\mathcal{B}$ are considered; and so on, until all the relevant arguments for and against are generated. These arguments can be arranged in the form of a tree that has $\mathcal{A}$ as its root, arcs are the defeat relation, and hence terminal nodes are unattacked arguments. At this point, $\mathcal{A}$ is assigned a label (undefeated, or defeated), according to a recursive labeling procedure in this tree of arguments. The procedure determines whether $\mathcal{A}$ is undefeated, i.e. whether it constitutes a solid justification or explanation for the truth of $\langle \ell, t \rangle$. In case it is, we say $\langle \ell, t \rangle$ is warranted in the knowledge base (Π, Δ).

For the temporal component, we take the set of natural numbers $\mathbb{N}$ as our working set of discrete time points. The logic t-DeLP is based on temporal literals $\langle \ell, t \rangle$, where ℓ is a literal and $t \in \mathbb{N}$, denoting ℓ *holds at time* t. In order to solve conflicts between arguments, the preference (or defeat) relation between arguments will be based on: a preference for arguments with *more premises* and *more recent* information. The latter notion of preference allows an argument $\mathcal{A}$ to defeat the persistence of previous steps in $\mathcal{A}$ that are not to persist according to $\mathcal{A}$ itself. In addition, since arguments must be consistent with strict information, strict arguments cannot be attacked. A further criterion, *less durative rules*, is not addressed here.[3]

Definition 1 (Literal, Rule) Given a finite set of propositional variables Var, we define $\mathsf{Lit} = \mathsf{Var} \cup \{\sim p \mid p \in \mathsf{Var}\}$. The set of temporal literals is defined as $\mathsf{TLit} = \{\langle \ell, t \rangle \mid \ell \in \mathsf{Lit}, t \in \mathbb{N}\}$. A *temporal defeasible (resp. strict) rule* is an expression δ relating temporal literals of the form

$$\langle \ell, t \rangle \prec \langle \ell_0, t_0 \rangle, \ldots, \langle \ell_n, t_n \rangle \qquad (\text{resp.} \quad \langle \ell, t \rangle \leftarrow \langle \ell_0, t_0 \rangle, \ldots, \langle \ell_n, t_n \rangle),$$

where $t \geq \max\{t_0, \ldots t_n\}$. We write $\mathsf{body}(\delta) = \{\langle \ell_0, t_0 \rangle, \ldots, \langle \ell_n, t_n \rangle\}$, $\mathsf{head}(\delta) = \langle \ell, t \rangle$ and $\mathsf{literals}(\delta) = \{\mathsf{head}(\delta)\} \cup \mathsf{body}(\delta)$.

A strict rule with an empty body, e.g. $\langle \ell, t \rangle \leftarrow$, also denoted $\langle \ell, t \rangle$, represents a basic fact that holds at time t. As in most of the DeLP literature -see [23], [13]- basic defeasible facts, or presumptions, of the form $\langle \ell, t \rangle \prec$, are not considered. A strict rule $\delta \in \Pi$ preserves the truth from $\mathsf{body}(\delta)$ to $\mathsf{head}(\delta)$. A defeasible rule $\delta \in \Delta$ states a weaker claim: if the premises are true, this is in principle a reason for believing that the conclusion is also true. This conclusion, though, may be later withdrawn when other reasons are considered.

[3] This is important, since rules with long duration might fail to detect conflicts, (so, e.g. the program might fail to predict that two balls running into each will collide). Instead, we will assume rules are precise enough for the problem at hand.

A special subset of (defeasible) rules is that of *persistence* rules, of the form $\langle \ell, t+1\rangle \prec \langle \ell, t\rangle$ and stating that, unless there exist reasons to the contrary, ℓ is preserved from t to $t+1$ (if true at t). The set of defeasible persistence rules will be denoted Δ_{p}.

Strict persistence rules and strict durative rules carry a strong commitment on the preservation of a literal, and they will not in general be considered, examples as follows notwithstanding.

Example 2 $\langle \sim\text{tuesday}, t+24\rangle \leftarrow \langle \text{tuesday}, t\rangle$ and $\langle \text{wednesday}, t+24\rangle \leftarrow \langle \text{tuesday}, t\rangle$, where time units are hours, are examples of strict durative rules about facts that are true by linguistic convention. Since literals as above will not contradicted by contingent rules, the previous rules can safely be modeled as defeasible rules, with $\prec$ replacing $\leftarrow$.

Definition 2 (Derivability, Consistent Set) Given a set of rules and strict facts Γ, we say a literal $\langle \ell, t\rangle$ *derives from* Γ, denoted $\Gamma \vdash \langle \ell, t\rangle$ or also $\langle \ell, t\rangle \in \mathrm{Cn}(\Gamma)$ iff

- $\langle \ell, t\rangle \in \Gamma$, or
- there exists $\delta \in \Gamma$ with $\mathsf{head}(\delta) = \langle \ell, t\rangle$, and such that $\mathsf{body}(\delta)$ is a set of literals that derive from Γ.

We say Γ is *consistent* iff the set $\mathrm{Cn}(\Gamma)$ contains no pair of literals of the form $\langle \ell, t\rangle$ and $\langle \sim\ell, t\rangle$. In particular, a set of literals is consistent iff it does not contain such a contradictory pair of literals $\langle \ell, t\rangle, \langle \sim\ell, t\rangle$.

Note that derivability is monotonic: $\mathrm{Cn}(\Gamma) \subseteq \mathrm{Cn}(\Gamma')$ whenever $\Gamma \subseteq \Gamma'$.

Definition 3 (Program) A *t-DeLP program*, or t-de.l.p., is a pair (Π, Δ) where $\Pi = \Pi_{\mathsf{f}} \cup \Pi_{\mathsf{r}}$ is a consistent set of temporal strict facts and rules, and Δ a set of temporal defeasible rules.

Temporal rules as above can be seen as instances of general rules δ^* of the form

$$\ell \prec (\ell_0, d_0), \ldots, (\ell_n, d_n)$$

-and similarly for strict rules with $\leftarrow$ -, where each d_i expresses how much time in advance must ℓ_i hold for the rule to apply and produce a derivation of ℓ. Such a general rule is to be understood as a shorthand for the set of rules

$$\{\langle \ell, t\rangle \prec \langle \ell_0, t-d_0\rangle, \ldots, \langle \ell_n, t-d_n\rangle \mid t \in \mathbb{N}, t \geq \max\{d_0, \ldots, d_n\}\}.$$

For example, the rule

$$\langle p, 4\rangle \prec \langle q, 3\rangle \text{ would be an instance of } p \prec (q, 1).$$

Persistence rules can therefore be expressed as general rules of the form $\ell \prec (\ell, 1)$; this defeasible general persistence rule for ℓ will be denoted δ_ℓ, and an instance $\langle \ell, t+1\rangle \prec \langle \ell, t\rangle$ of δ_ℓ will also be denoted by $\delta_\ell(t)$; similarly, a set of instances of δ_ℓ given by an interval $[t, \ldots, t+k]$ will be denoted $\{\delta_\ell(t')\}_{t \leq t' \leq t+k}$.

Though the general rules notation becomes handy at some points through the paper, the formal definitions below do make use only of temporal rules, i.e. instances of the above. Unless stated otherwise, in the remaining of the paper we will mean by *rule* an expression as in Definition 1 that is not a fact (i.e. with non-empty body).

Example 3 Consider the situation described next and formalized in Figure 3. Lars, a tourist visiting the Snake Forest, has just been bitten by a venomous snake (denoted $@\mathsf{forest}(\mathrm{Lars})$, and $\mathsf{bitten}^*(\mathrm{Lars})$. [4] The poison of this type of snake does kill a person in 3 hours (δ_1). But since our subject, Lars, is experienced (it has been bitten and cured a few times before), denoted $\mathsf{exp}(\mathrm{Lars})$, he may resist up to 5 hours (δ_2, δ_3). We decide to take him to the nearest hospital. In normal conditions this would take 2 hours (δ_4), but since today is sunday, the traffic jam (δ_7) makes it impossible to reach the hospital in less than 4 hours (δ_5, δ_6). The antidote takes less than an hour to become effective (δ_8), and is given to persons that are at the hospital, have been recently bitten (denoted $\mathsf{bitten}(\cdot)$) and are alive (denoted $\sim\mathsf{dead}(\cdot)$). We prove below in t-DeLP that Lars survives the snake attack.

$$\Pi$$

$$\left\{\begin{array}{lll} \langle @\mathsf{forest}(\mathrm{Lars}), 0\rangle, & \langle \mathsf{bitten}^*(\mathrm{Lars}), 0\rangle, & \langle \mathsf{exp}(\mathrm{Lars}), 0\rangle, \\ \langle \sim\mathsf{dead}(\mathrm{Lars}), 0\rangle, & \langle \mathsf{sunday}, 0\rangle & \end{array}\right\}$$

$$\Delta$$

$\mathsf{bitten}(\mathrm{Lars})$	$\prec$	$\langle \mathsf{bitten}^*(\mathrm{Lars}), 0\rangle$	δ_0
$\mathsf{dead}(\mathrm{Lars})$	$\prec$	$(\mathsf{bitten}^*(\mathrm{Lars}), 3)$	δ_1
$\sim\mathsf{dead}(\mathrm{Lars})$	$\prec$	$(\mathsf{bitten}^*(\mathrm{Lars}), 3), (\mathsf{exp}(\mathrm{Lars}), 3), (\sim\mathsf{dead}(\mathrm{Lars}), 3)$	δ_2
$\mathsf{dead}(\mathrm{Lars})$	$\prec$	$(\mathsf{bitten}^*(\mathrm{Lars}), 5), (\mathsf{exp}(\mathrm{Lars}), 5), (\sim\mathsf{dead}(\mathrm{Lars}), 5)$	δ_3
$@\mathsf{hospital}(\mathrm{Lars})$	$\prec$	$(\mathsf{bitten}^*(\mathrm{Lars}), 2), (@\mathsf{forest}(\mathrm{Lars}), 2), (\sim\mathsf{dead}(\mathrm{Lars}), 2)$	δ_4
$\sim@\mathsf{hospital}(\mathrm{Lars})$	$\prec$	$\left\{\begin{array}{ll} (\mathsf{traffic.jam}, 2), & (\mathsf{bitten}^*(\mathrm{Lars}), 2), \\ (\sim\mathsf{dead}(\mathrm{Lars}), 2), & (@\mathsf{forest}(\mathrm{Lars}), 2) \end{array}\right\}$	δ_5
$@\mathsf{hospital}(\mathrm{Lars})$	$\prec$	$\left\{\begin{array}{ll} (\mathsf{traffic.jam}, 4), & (\mathsf{bitten}^*(\mathrm{Lars}), 4), \\ (\sim\mathsf{dead}(\mathrm{Lars}), 4), & (@\mathsf{forest}(\mathrm{Lars}), 4) \end{array}\right\}$	δ_6
$\mathsf{traffic.jam}$	$\prec$	$(\mathsf{sunday}, 0)$	δ_7
$\sim\mathsf{dead}(\mathrm{Lars})$	$\prec$	$(@\mathsf{hospital}(\mathrm{Lars}), 1), (\mathsf{bitten}(\mathrm{Lars}), 1), (\sim\mathsf{dead}(\mathrm{Lars}), 1)$	δ_8
		plus $\delta_\ell \in \Delta_\mathsf{p}$ for each $\ell \notin \{\mathsf{bitten}^*(\mathrm{Lars}), \sim@\mathsf{loc}(\mathrm{Lars})\}$	δ_ℓ

Fig. 1 The list of strict facts, defeasible rules δ_1-δ_8 and persistence rules δ_ℓ for Example 3.

As it happens in DeLP, the set of derivable literals in (Π, Δ) will not in general be consistent. The first step to enforce consistency is to focus on those derivations that have the form of an argument.

Definition 4 (Argument) Given a t-de.l.p. (Π, Δ), an *argument* for $\langle \ell, t\rangle$ is a set $\mathcal{A} = \mathcal{A}_\Pi \cup \mathcal{A}_\Delta$, with $\mathcal{A}_\Pi \subseteq \Pi$ and $\mathcal{A}_\Delta \subseteq \Delta$, such that:

(1) $\mathcal{A}_\Delta \cup \Pi \vdash \langle \ell, t\rangle$,
(2) $\Pi \cup \mathcal{A}_\Delta$ is consistent,
(3) $\mathcal{A}_\Delta$ is $\subseteq$-minimal satisfying (1) and (2).
(4) $\mathcal{A}_\Pi$ is $\subseteq$-minimal satisfying $\mathcal{A}_\Delta \cup \mathcal{A}_\Pi \vdash \langle \ell, t\rangle$

[4] We use two literals $\mathsf{bitten}^*(\cdot)$ and $\mathsf{bitten}(\cdot)$. The literal with an asterisk is used to track the (unique) time where the snake bite occurred, and hence will not be allowed to persist (i.e. no persistence rules for this literal will exist in the program). The second literal $\mathsf{bitten}(\cdot)$ just denotes the fact of *having been (recently) bitten* and persistence rules for it are assumed.

Thus, arguments are consistent minimal derivations (i.e. without redundant information) and using minimal defeasible information. In particular, if a strict argument exists for some literal, then no defeasible derivation for the same literal constitutes an argument. In Example 3, each possible argument consists of facts in Π_{f} and rules in Δ. Observe that, although Π and Δ may be infinite (due to the coding of general rules as an infinite set of temporal rules), an argument for a t-de.l.p. (Π, Δ) will always be a finite subset of $\Pi \cup \Delta$. Given an argument $\mathcal{A}$ for $\langle \ell, t \rangle$, we also define:

$$\begin{array}{rclrcl} \mathsf{concl}(\mathcal{A}) & = & \langle \ell, t \rangle & \mathsf{base}(\mathcal{A}) & = & \mathsf{body}[\mathcal{A}] \smallsetminus \mathsf{head}[\mathcal{A}] \\ \|\mathcal{A}\| & = & t - t(\mathcal{A}) & \mathsf{literals}(\mathcal{A}) & = & (\bigcup \mathsf{body}[\mathcal{A}]) \cup \mathsf{head}[\mathcal{A}] \end{array}$$

where $t(\mathcal{A}) = \min\{t' \in \mathbb{N} \mid \langle \cdot, t' \rangle \in \mathsf{base}(\mathcal{A})\}$.

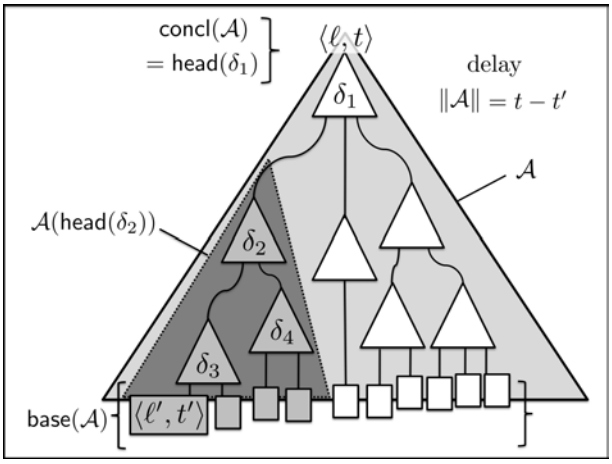

Fig. 2 The internal structure of an argument $\mathcal{A}$, as a set of facts (rectangles) and rules (small triangles). The dotted triangle represents the sub-argument $\mathcal{A}(\mathsf{head}(\delta_2))$ of $\mathcal{A}$ determined by the literal $\mathsf{head}(\delta_2)$. The maximum difference between time-points of literals in $\mathcal{A}$, i.e. $t - t'$, defines the delay of $\mathcal{A}$.

In DeLP, arguments are defined as sets of defeasible rules $\mathcal{A} \subseteq \Delta$, leaving open how these are to be completed by Π to obtain a (minimal, consistent) derivation of some literal $\langle \ell, t \rangle$; since different completions allow for different conclusions, one must make explicit which is the intended conclusion (e.g. $\langle \mathcal{A}, \ell \rangle$). In contrast, we explicitly fix in an argument $\mathcal{A}$ which are its strict rules, hence making the conclusion $\mathsf{concl}(\mathcal{A})$ uniquely determined by $\mathcal{A}$. This definition simplifies the detection of inconsistencies with intermediate steps in the strict part of $\mathcal{A}$. With more detail, there can be several ways to complete defeasible rules in $\mathcal{A}$ into a derivation for $\mathsf{concl}(\mathcal{A})$, and each of them can be attacked by different arguments. For example, the sets

$$\left\{ \begin{array}{c} \langle p, 4 \rangle \leftarrow \langle q, 2 \rangle \\ \langle q, 2 \rangle \leftarrow \langle r, 1 \rangle \\ \langle r', 0 \rangle \end{array} \right\} \quad \text{and} \quad \left\{ \begin{array}{c} \langle p, 4 \rangle \leftarrow \langle s, 3 \rangle \\ \langle s, 3 \rangle \leftarrow \langle r, 1 \rangle \\ \langle r', 0 \rangle \end{array} \right\}$$

may both complete the set of defeasible rules $\{\langle p', 5 \rangle \prec \langle p, 4 \rangle, \langle r, 1 \rangle \prec \langle r', 0 \rangle\} \subseteq \Delta$ into an argument (derivation) for $\langle p', 5 \rangle$, but only the latter is attacked by an argument concluding $\langle \sim s, 3 \rangle$.

Now we define a sub-argument of an argument $\mathcal{A}$. A sub-argument will be the actual target of an attack by another argument.

Definition 5 (Sub-argument) Let (Π, Δ) be a t-de.l.p. and let $\mathcal{A}$ be an argument for $\langle \ell, t \rangle$ in (Π, Δ). Given some $\langle \ell_0, t_0 \rangle \in \mathsf{literals}(\mathcal{A})$, a *sub-argument* for $\langle \ell_0, t_0 \rangle$ is a subset $\mathcal{B} \subseteq \mathcal{A}$ such that $\mathcal{B}$ is an argument for $\langle \ell_0, t_0 \rangle$.

The inductive definition for computing the sub-argument induced by some literal is straightforward (see the Appendix).

Proposition 1 *Given some argument $\mathcal{A}$ and a literal $\langle \ell, t \rangle \in \mathsf{literals}(\mathcal{A})$, then the sub-argument of $\mathcal{A}$ for $\langle \ell, t \rangle$ is unique.*

From here on, this unique sub-argument of $\mathcal{A}$ induced by $\langle \ell_0, t_0 \rangle$ will be denoted $\mathcal{A}(\langle \ell_0, t_0 \rangle)$. For example, in Figure 2, $\mathcal{A}(\mathsf{head}(\delta_2)) = \{\delta_2, \delta_3, \delta_4, \langle \ell', t' \rangle, \ldots\}$.

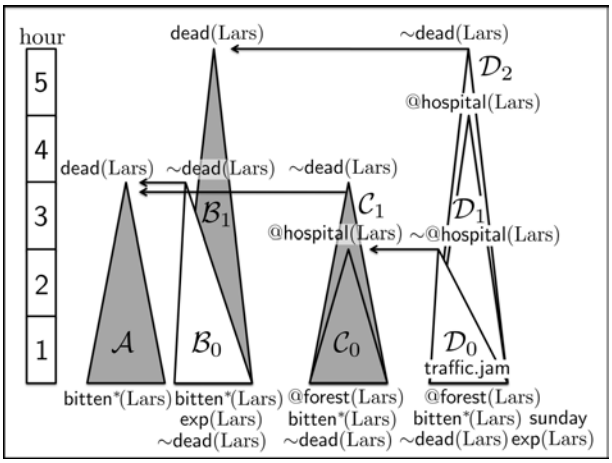

Fig. 3 The Snake Bites Lars scenario. Arguments are depicted as triangles, with arrows denoting conflicts among them. Arguments for which defeaters exist are depicted in grey.

Definition 6 (Attack) Given a t-de.l.p. (Π, Δ), let $\mathcal{A}_0$ and $\mathcal{A}_1$ be arguments. We say $\mathcal{A}_1$ *attacks* $\mathcal{A}_0$ iff $\sim\mathsf{concl}(\mathcal{A}_1) \in \mathsf{literals}[\mathcal{A}_0]$, where we use the notation $\sim\langle \ell, t \rangle$ to denote $\langle \sim\ell, t \rangle$. In this case, we also say that $\mathcal{A}_1$ attacks $\mathcal{A}_0$ at the sub-argument $\mathcal{A}_0(\sim\mathsf{concl}(\mathcal{A}_1))$.

Notice that an argument $\mathcal{A}_1$ cannot attack another $\mathcal{A}_0$ at a sub-argument consisting of strict information only (i.e. if $\mathcal{A}_0(\sim\mathsf{concl}(\mathcal{A}_1)) \subseteq \Pi$), since in this case $\mathcal{A}_1$ would not be consistent with Π, and hence $\mathcal{A}_1$ would not even be an argument.

As in DeLP, one refines the relation of attack relation into a defeat relation to decide which argument prevails in case of an attack. This relation could be in principle specified by the user[5], but in this paper we adopt a formal criterion in order to meet the intuitive preferences exemplified next.

[5] See [23] for a procedure based on a preference relation between rules. From this relation, one can induce a preference between any two arguments, by comparing each rule $\delta \in \mathcal{A}$ with each rule $\delta' \in \mathcal{B}$.

Example 4 See Figure 3 for an illustration of Example 3. The arguments are defined by the following rules (facts are not listed here):

$$\mathcal{A} \supseteq \{\delta_1(0)\} \qquad \mathcal{B}_0 \supseteq \{\delta_2(0)\} \qquad \mathcal{B}_1 \supseteq \{\delta_3(0)\} \qquad \mathcal{C}_0 \supseteq \{\delta_4(0)\}$$
$$\mathcal{C}_1 \supseteq \{\delta_4(0), \delta_8(2)\} \cup \{\delta_0(t), \delta_{\mathsf{bitten}(\mathrm{Lars})}(t), \delta_{\sim\mathsf{dead}(\mathrm{Lars})}(t)\}_{0\leq t<2}$$
$$\mathcal{D}_0 \supseteq \{\delta_7(0), \delta_5(0)\} \qquad \mathcal{D}_1 \supseteq \mathcal{B}_0 \cup \{\delta_{\sim\mathsf{dead}(\mathrm{Lars})}(3), \delta_7(0), \delta_6(0)\}$$
$$\mathcal{D}_2 \supseteq \mathcal{D}_1 \cup \{\delta_0(0), \delta_8(4)\} \cup \{\delta_{\mathsf{bitten}(\mathrm{Lars})}(t)\}_{0\leq t\leq 3}$$

The arguments related by an arrow attack each other: $\mathcal{C}_1$, $\mathcal{B}_0$ attack $\mathcal{A}$ and viceversa. But there are asymmetries in the quantity of information supporting each argument. Intuitively, in this example we have:

(1) $\mathcal{B}_0$ should prevail over $\mathcal{A}$ since it is based on more information (the premises of $\mathcal{A}$ are a proper subset of those in $\mathcal{B}_0$); such an asymmetry between $\mathcal{B}_0$ and $\mathcal{A}$ makes the latter not to count as a reason against $\mathcal{B}_0$. (See also Figure 3.)

To illustrate another kind of asymmetry in the amount of information, consider a new example:

(2) Suppose you hold an object o at some distance d_0 from the floor, and drop it at $t = 0$. It is expected to crash into the floor at, say, $t = 3$. This is modeled by an argument $\mathcal{A}$ having $\mathsf{base}(\mathcal{A}) = \{\langle @(o, d_0), 0\rangle\}$, intermediate steps $\langle @(o, d_1), 1\rangle$ and $\langle @(o, d_2), 2\rangle$ (i.e. both in $\mathsf{literals}(\mathcal{A})$ given by appropriate rules), and conclusion $\mathsf{concl}(\mathcal{A}) = \langle @(o, 0), 3\rangle$; this latter literal $@(o, 0)$ denotes *o is at the floor*. Now, admitting (as we do) persistence rules for positive facts like $@(o, \cdot)$, an argument $\mathcal{B}$ can be constructed for the conclusion that the object will remain floating over the floor, e.g. at d_1. Namely, let $\mathcal{B} = \mathcal{A}(\langle @(o, d_1), 1\rangle) \cup \{\delta(t)_{@(o,d_1)}\}_{1\leq t<3}$. Note that while the asymmetry for case (1) above is missing $\mathsf{base}(\mathcal{B}) = \mathsf{base}(\mathcal{A})$, the argument $\mathcal{B}$ should not be a defeater for $\mathcal{A}$. The idea is that the existence of some reason for a change (like gravity w.r.t. the position of o) should override the use of persistence. Note that, after this change ceases to apply (e.g. when o lies at the floor), it is reasonable to use persistence to keep inferring that the position of o is the same, so far no further changes are known.

Finally, to illustrate blocking defeaters, rephrase Example 3 with these rules: black-spotted snakes are generally poisonous, while green snakes are generally harmless.

(3) If *a green black-spotted snake bites Lars*, we are not able to decide whether he has been poisoned, since reasons for and against do not dominate each other.

Definition 7 (Defeat) Let $\mathcal{A}_1$ attack $\mathcal{A}_0$ at $\mathcal{B}$, where $\mathsf{concl}(\mathcal{A}_1) = \langle \sim\ell, t\rangle$. We say $\mathcal{A}_1$ is a *proper defeater for* $\mathcal{A}_0$, denoted $\mathcal{A}_1 \succ \mathcal{A}_0$, iff

- $\mathsf{base}(\mathcal{A}_1) \supsetneq \mathsf{base}(\mathcal{B})$, or
- $\mathcal{B} = \mathcal{A}_1(\langle \ell, t'\rangle) \cup \{\delta_\ell(t'')\}_{t'\leq t''<t}$, for some $t' < t$.

We say $\mathcal{A}_1$ is a *blocking defeater for* $\mathcal{A}_0$ when $\mathcal{A}_1$ attacks $\mathcal{A}_0$ but $\mathcal{A}_1 \not\succ \mathcal{A}_0$ and $\mathcal{A}_0 \not\succ \mathcal{A}_1$. Blocking defeat relations are denoted $\mathcal{A}_1 \prec\!\succ \mathcal{A}_0$. Finally, a *defeater* is a proper or a blocking defeater.

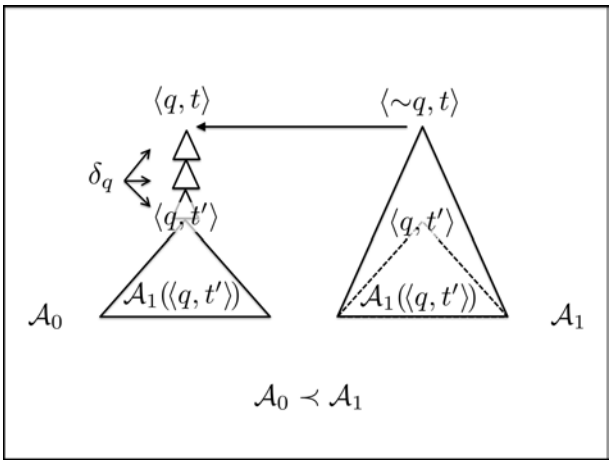

Fig. 4 An illustration of the second condition for $\mathcal{A}_1$ *is a proper defeater for* $\mathcal{A}_0$. The argument $\mathcal{A}_0$ (left) is obtained by extending a sub-argument $\mathcal{A}_1(\langle \ell, t'\rangle)$ of $\mathcal{A}_1$ (right) just with persistence rules δ_q from t' to t.

That, intuitively, the two criteria for proper defeat in Def. 7 suit the purposes of this paper is shown, resp., by Examples 3 (1)-(2). Example 3 (3) illustrates one of the cases of blocking defeat. In Definition 7, we adapt the formal criterion of *generalized specificity* [49] (widely used in DeLP) to causal reasoning with temporal literals so that the amount of information in an argument is measured by its initial causes[6] and by its use of substantive vs persistence rules (see Section 7.2 for more details on this issue).

Note that if we have $\mathcal{A}_1 \succ \mathcal{A}_0$ due to the second possibility, namely $\mathcal{A}_0 \supseteq \mathcal{B} = \mathcal{A}_1(\langle \ell, t'\rangle) \cup \{\delta_\ell(t'')\}_{t' \leq t'' < t}$, then $\mathsf{base}(\mathcal{B}) = \mathsf{base}(\mathcal{A}_1(\langle \ell, t'\rangle))$. In this second case, we will also say that $\mathcal{A}_1$ is (informationally) longer than $\mathcal{B}$.

As a result of this definition, any temporal change in truth-values, say from $\langle \sim p, 2\rangle$ to $\langle p, 5\rangle$, must accommodate an explanation for the non-persistence of the earlier $\sim p$, as occurring between 3 and 5 (if such persistence rules exist for $\sim p$). This accommodation may consist in the first criteria on bases of the arguments for $\langle p, 5\rangle$ and $\langle \sim p, 2\rangle$; or that the persistence of $\sim p$ is the only reason for $\langle p, 5\rangle$, given $\langle p, 2\rangle$; see $\mathcal{B}$ in Fig. 5 (Top). Otherwise, it can be easily defeated; see $\mathcal{A}_1$ in Fig. 5 (Bottom).

Proposition 2 *The following hold for any t-DeLP program:*

(1) If $\mathcal{A}_1$ is a proper defeater for an argument $\mathcal{A}_0$ at $\mathcal{B}$, then $\mathcal{B}$ is not a defeater for $\mathcal{A}_1$.
(2) If $\mathcal{A}, \mathcal{B}$ attack each other, and $\mathcal{B}$ is not a proper defeater for $\mathcal{A}$, then $\mathcal{A}$ is a defeater for $\mathcal{B}$.

An argument $\mathcal{B}$ defeating $\mathcal{A}$ can in its turn have its own defeaters $\mathcal{C}, \ldots$ and so on. (This is the case of $\mathcal{A}, \mathcal{B}_0, \mathcal{C}_0$ in Figure 3.) This gives rise to *argumentation lines* where each argument defeats its predecessor. Argumentation lines, though,

[6] Using the terminology of DeLP, the only *activation sets* of literals considered for comparing arguments are their respective bases.

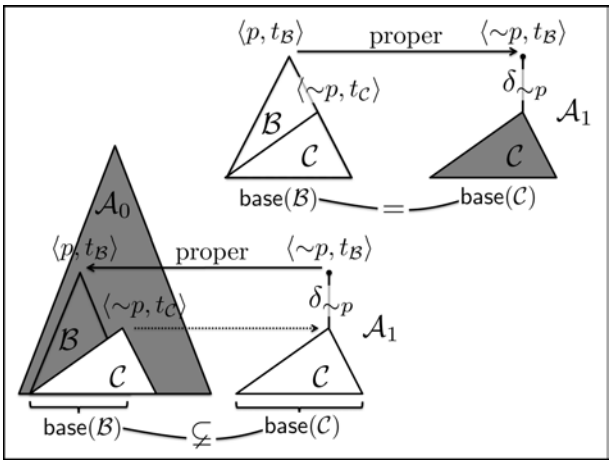

Fig. 5 Explanations of p that accommodate (Top), or not (Bottom) a previous instance of a $\sim p$ fact.

are not simply the composition of the defeat relation: again we refine this composition by imposing some further constraints. These constraints are needed to enforce desirable properties: finite length, acyclicity, and intuitive defense relations (counterattacks). For instance, in an argumentation line $[\ldots, \mathcal{A}, \mathcal{B}, \mathcal{C}, \ldots]$ we exclude the case where $\mathcal{C}$ is a blocking defeater for $\mathcal{B}$, *provided that* $\mathcal{B}$ is already blocking defeater for $\mathcal{A}$. This prevents the case $[\ldots, \mathcal{A}, \mathcal{B}, \mathcal{A}, \ldots]$. Other forms of cyclic defeats $[\ldots, \mathcal{A}, \mathcal{B}, \ldots, \mathcal{A}, \mathcal{B}, \ldots]$ are also excluded in the definition. The following definition is adapted from [23] to the present framework.

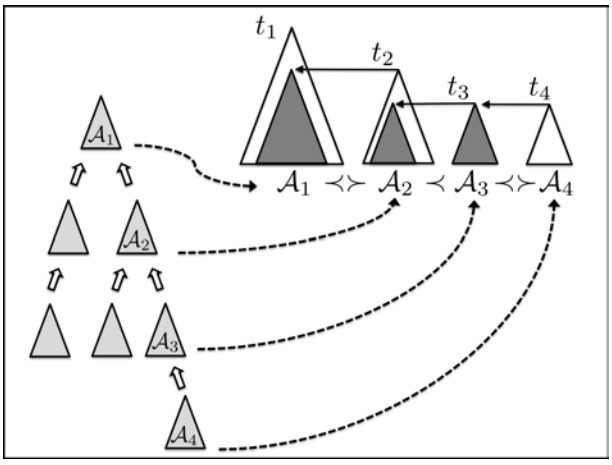

Fig. 6 (Right) An argumentation line $\Lambda = [\mathcal{A}_1, \ldots, \mathcal{A}_4]$, with defeated sub-arguments depicted in grey. Notice that the time of these attacks is decreasing, and that condition (iii) from Def. 8 is satisfied. (Left) The same argumentation line Λ is depicted as part of the dialectical tree $\mathcal{T}_{(\Pi,\Delta)}(\mathcal{A}_1)$.

Definition 8 (Argumentation Line, Dialectical Tree) Let $\mathcal{A}_1$ be an argument in (Π, Δ). An *argumentation line* for $\mathcal{A}_1$ is a sequence $\Lambda = [\mathcal{A}_1, \mathcal{A}_2, \ldots]$ where

(i) supporting arguments, i.e. those in odd positions $\mathcal{A}_{2i+1} \in \Lambda$ are jointly consistent with Π, and similarly for interfering arguments $\mathcal{A}_{2i} \in \Lambda$
(ii) a supporting (interfering) argument is different from the attacked sub-arguments of previous supporting (interfering) arguments: $\mathcal{A}_{i+2k} \neq \mathcal{A}_i(\sim\mathsf{concl}(\mathcal{A}_{i+1}))$.
(iii) $\mathcal{A}_{i+1}$ is a proper defeater for $\mathcal{A}_i$ if $\mathcal{A}_i$ is a blocking defeater for $\mathcal{A}_{i-1}$

An argumentation line $[\mathcal{A}_1, \ldots, \mathcal{A}_n]$ for $\mathcal{A}_1$ is *maximal* if there is no other argument $\mathcal{A}_{n+1}$ such that $[\mathcal{A}_1, \ldots, \mathcal{A}_n, \mathcal{A}_{n+1}]$ is an arg. line for $\mathcal{A}_1$.

It is clear that the set of maximal argumentation lines for $\mathcal{A}_1$ can be arranged in the form of a tree, where all paths from the root $\mathcal{A}_1$ to the leaf nodes exactly correspond to all the possible maximal argumentation lines for $\mathcal{A}_1$. This tree is called *dialectical tree* for $\mathcal{A}_1$, and is denoted $\mathcal{T}_{(\Pi,\Delta)}(\mathcal{A}_1)$ (see [23] for more details). We will also express that a sequence of arguments $\Lambda = [\mathcal{A}_1, \ldots]$ is a (non-necessarily maximal) argumentation line for $\mathcal{A}_1$ by $\Lambda \in \mathcal{T}_{(\Pi,\Delta)}(\mathcal{A}_1)$.

Remark 1 While (i) and (iii) are exactly as in DeLP, the above condition (ii) is less restrictive than its counterpart in [23]. In this work, a sub-argument of $\mathcal{A}_i$ cannot (indirectly) defend this argument. That is, a sub-argument of $\mathcal{A}_i$ cannot occur as $\mathcal{A}_{i+2j}$ in the same argumentation line. In our temporal case, we adopt a more liberal view concerning defenses based on sub-arguments: for instance, a sub-argument talking about a previous time might offer legitimate reasons to the defense of $\mathcal{A}_i$. If its only available defense was in this sense a proper part of the attacked argument, then it should be admitted. (See the next example.)

Example 5 We expand the arguments of Example 4 (2) as follows: suppose an object o will fall from height or distance d_0 into the floor, denoted by distance 0. That is, a transition from $\langle @(o, d_0), \cdot\rangle$ to $\langle @(o, 0), \cdot\rangle$ will happen. Moreover, assume that o is an egg, and also that a boiling pot of water is awaiting at the floor. The temperature at d_1 is cold (i.e. not hot). Thus, we have

$$\Pi_{\mathsf{f}} = \{\langle @(o, d_0), 0\rangle, \langle \mathsf{hot}(0), 0\rangle, \langle \sim\mathsf{hot}(d_1), 0\rangle, \langle \sim\mathsf{boils}(o), 0\rangle\}$$

The set Δ is as before in Example 4 (2), plus persistence rules for local heat or coldness: $\delta_{\mathsf{hot}(0)}, \delta_{\sim\mathsf{hot}(d_1)}$, and also the heat-transfer rules

$$\begin{array}{lrcl} \delta_1(t): & \langle \mathsf{boils}(o), t+1\rangle & \prec & \langle \mathsf{hot}(0), t\rangle, \langle @(o, 0), t\rangle \\ \delta_2(t): & \langle \sim\mathsf{boils}(o), t+1\rangle & \prec & \langle \sim\mathsf{hot}(d_1), t\rangle, \langle @(o, d_1), t\rangle \end{array}$$

The arguments to conclude that the egg will ($\mathcal{A}^+$) or will not ($\mathcal{B}^+$, among others) boil at $t = 4$ are defined as

$$\begin{array}{rcl} \mathcal{A}^+ & = & \mathcal{A} \cup \{\langle \mathsf{hot}(0), 0\rangle\} \cup \{\delta(t)_{\mathsf{hot}(0)}\}_{0\leq t<3} \cup \{\delta_1(3)\} \\ \mathcal{B}^+ & = & \mathcal{B} \cup \{\langle \sim\mathsf{hot}(d_1), 0\rangle\} \cup \{\delta(t)_{\sim\mathsf{hot}(d_1)}\}_{0\leq t<3} \cup \{\delta_2(3)\} \end{array}$$

where $\mathcal{A}$ and $\mathcal{B}$ are as in Example 4 (2). We have $\mathsf{concl}(\mathcal{A}^+) = \langle \mathsf{boils}(o), 4\rangle$ and $\mathsf{concl}(\mathcal{B}^+) = \langle \sim\mathsf{boils}(o), 4\rangle$, so $\mathcal{A}^+$ and $\mathcal{B}^+$ attack each other, just like $\mathcal{A}$ and $\mathcal{B}$. But now, these $\mathcal{A}^+$ and $\mathcal{B}^+$ express: the expected fall-and-boiling of the egg ($\mathcal{A}$) and,

respectively, that the egg keeps floating in air and stays unboiled ($\mathcal{B}$). A more relevant difference with the pair $\mathcal{A}, \mathcal{B}$ is that $\mathcal{A}^+$ is not longer than $\mathcal{B}^+$. In fact, we need $\mathcal{A}$ to defeat $\mathcal{B}^+$ at $\mathcal{B}$. Thus, Definition 8 allows for $[\mathcal{A}^+, \mathcal{B}^+, \mathcal{A}]$ to be an arg. line, so $\mathcal{A}$ can defend $\mathcal{A}^+$. If these defending sub-arguments were not allowed (see Definition 9 below), we could not conclude that the egg becomes hot at $t = 4$.

Lemma 1 *For any t-de.l.p.* (Π, Δ),

(1) If $[\mathcal{A}_1, \ldots, \mathcal{A}_m, \ldots, \mathcal{A}_n]$ *is an argumentation line for* $\mathcal{A}_1$*, then* $[\mathcal{A}_m, \ldots, \mathcal{A}_n]$ *is an argumentation line for* $\mathcal{A}_m$*.*
(2) Each argumentation line $\Lambda = [\mathcal{A}_1, \ldots] \in \mathcal{T}_{(\Pi,\Delta)}(\mathcal{A}_1)$ *is finite. The dialectical tree* $\mathcal{T}_{(\Pi,\Delta)}(\mathcal{A}_1)$ *is finite.*

The following definitions of the marking procedure of dialectical trees and the notion of warrant exactly follow those of DeLP.

Definition 9 (Marking) Let $\mathcal{T} = \mathcal{T}_{(\Pi,\Delta)}(\mathcal{A}_1)$ be the dialectical tree for $\mathcal{A}_1$. Then,

(1) mark all terminal nodes of $\mathcal{T}$ with a U (for undefeated);
(2) mark a node $\mathcal{B}$ with a D (for defeated) if it has a children node marked U;
(3) mark $\mathcal{B}$ with U if all its children nodes are marked D.

Initially all the arguments in the dialectical tree $\mathcal{T}_{(\Pi,\Delta)}(\mathcal{A}_1)$ are unmarked (grey) as in Figure 6 (Left). To illustrate the marking procedure, see Figure 7, where arguments marked U are represented white, and those marked D are represented black.

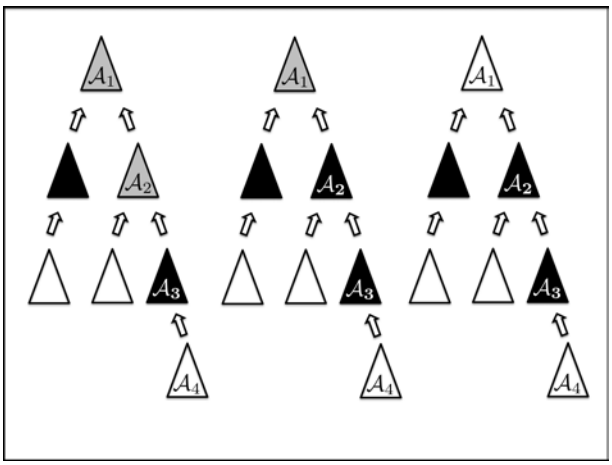

Fig. 7 The marking procedure in the dialectical tree for an argument $\mathcal{A}_1$: (Left) terminal nodes, e.g. $\mathcal{A}_4$, are marked undefeated (white) first. (All) If a node, e.g. $\mathcal{A}_3$, has an undefeated child it is marked defeated (black). If, otherwise, all their children are marked defeated this node is marked undefeated, e.g. $\mathcal{A}_1$. (Right) The procedure ends with the root argument $\mathcal{A}_1$ being marked, undefeated in this case.

Note that in a dialectical tree $\mathcal{T}_{(\Pi,\Delta)}(\mathcal{A}_1)$, an argument $\mathcal{A}$ can occur in different positions of several (maximal) argumentation lines in $\Lambda, \Lambda', \ldots \in \mathcal{T}_{(\Pi,\Delta)}(\mathcal{A}_1)$. In

this case, the marking of $\mathcal{A}$ in Λ can be different from the marking of $\mathcal{A}$ in Λ'. Given an argumentation line $\Lambda = [\mathcal{A}_1, \mathcal{A}_2, \mathcal{A}_3, \ldots, \mathcal{A}_n] \in \mathcal{T}_{(\Pi,\Delta)}(\mathcal{A}_1)$, we will express the evaluation of its arguments along Λ according to the marking procedure by a corresponding sequence of D's and U's , e.g. $[D, D, U, \ldots, U]$.

Definition 10 (Warrant) Given a t-de.l.p. (Π, Δ), we say $\langle \ell, t \rangle$ is *warranted* in (Π, Δ) iff there exists an argument $\mathcal{A}_1$ for $\langle \ell, t \rangle$ in (Π, Δ) such that $\mathcal{A}_1$ is undefeated, i.e. marked U, in $\mathcal{T}_{(\Pi,\Delta)}(\mathcal{A}_1)$. We will denote by $\mathsf{warr}(\Pi, \Delta)$ the set of warranted literals in (Π, Δ).

In the particular case of strict arguments $\mathcal{A} \subseteq \Pi$, we will have that $\mathcal{T}_{(\Pi,\Delta)}(\mathcal{A})$ only contains the argumentation line $[\mathcal{A}]$, so each strictly derivable fact is warranted. For any other argument $\mathcal{B}$, the strict argument $\mathcal{A}$ cannot occur in any argumentation line in $\mathcal{T}_{(\Pi,\Delta)}(\mathcal{B})$.

Example 6 (Cont'd) Recall Example 3. The arguments in Fig. 3 related by an arrow stand in the relation of proper defeat, e.g. $\mathcal{A} \leftarrow \mathcal{B}_0$ denotes $\mathcal{B}_0$ is a proper defeater for $\mathcal{A}$. Thus we have the dialectical trees for each argument consist of the following argumentation lines (with the corresponding evaluations):

$$\begin{array}{rcll}
\mathcal{T}_{(\Pi,\Delta)}(\mathcal{A}) & = & \left\{\begin{array}{l}[\mathcal{A}, \mathcal{B}_0], \\ [\mathcal{A}, \mathcal{C}_1, \mathcal{D}_0]\end{array}\right\} & \begin{array}{l}[D, U], \\ [D, D, U]\end{array} \\
\mathcal{T}_{(\Pi,\Delta)}(\mathcal{B}_0) & = & \{[\mathcal{B}_0]\} & [U] \\
\mathcal{T}_{(\Pi,\Delta)}(\mathcal{B}_1) & = & \{[\mathcal{B}_1, \mathcal{D}_2]\} & [D, U] \\
\mathcal{T}_{(\Pi,\Delta)}(\mathcal{C}_i) & = & \{[\mathcal{C}_i, \mathcal{D}_0]\} & [D, U], \text{ for each } i \in \{0, 1\} \\
\mathcal{T}_{(\Pi,\Delta)}(\mathcal{D}_j) & = & \{[\mathcal{D}_j]\} & [U], \text{ for each } j \in \{0, 1, 2\}
\end{array}$$

Since $\mathcal{D}_2$ is undefeated, we (defeasibly) conclude that Lars will be alive at $t = 5$.

Let us now solve scenario (2) from Example 4. Recall the argument $\mathcal{A}$ concluding that dropping the object will indeed cause it to crash into the floor (distance 0) at $t = 3$. $\mathcal{A}$ is a proper defeater for the rival arguments stating that the object will keep floating in the air once it reaches distance d_0, d_1 or d_2. Call these arguments $\mathcal{B}_0, \mathcal{B}_1$ and $\mathcal{B}_2$, resp. (Incidentally, $\mathcal{B}_1$ properly defeats $\mathcal{B}_0$ and so does $\mathcal{B}_2$ with $\mathcal{B}_0, \mathcal{B}_1$.) If these arguments capture all the relevant phenomena in this scenario, then $\mathcal{T}_{(\Pi,\Delta)}(\mathcal{A}) = [\mathcal{A}]$ so $\mathcal{A}$ is undefeated; on the other hand, for any $0 \leq i < 3$ the dialectical tree $\mathcal{T}_{(\Pi,\Delta)}(\mathcal{B}_i)$ contains a maximal argumentation line $\Lambda = [\mathcal{B}_i, \mathcal{A}]$, among possibly others. This can only be evaluated as $[D, U]$, so each $\mathcal{B}_i$ is defeated.

For another example more in line with the AI tradition, consider the well-known Yale shooting scenario, where a turkey is expected to be killed, after someone loads a gun and shoots at it, possibly with some *waiting* actions taking place before or between these two actions. Early logical formalisms were able to prove that the turkey dies *if shot just after loading the gun*, while failed to prove the same if some waiting occurred between load and shoot. This prompted the use of frame axioms to describe what persists after some change. In t-DeLP, we proposed instead formal criteria to decide about particular uses of persistence.

Example 7 (Yale Shooting Scenario) In t-DeLP, we can solve this scenario by representing:

$$\text{action shoot (at } t) \quad \text{by rules:} \quad \begin{array}{l} \delta_0(t) = \langle \sim\mathsf{loaded}, t+1\rangle \prec \langle \mathsf{loaded}, t\rangle, \\ \delta_1(t) = \langle \sim\mathsf{alive}, t+1\rangle \prec \langle \mathsf{loaded}, t\rangle; \end{array}$$

$$\text{action load (at } t) \quad \text{by rule:} \quad \delta(t) = \langle \mathsf{loaded}, t+1\rangle \prec \langle \sim\mathsf{loaded}, t\rangle.$$

Instead of having an explicit wait action, we just let persistence rules δ_ℓ apply for any literal, i.e. any $\ell \in \{\mathsf{loaded}, \sim\mathsf{loaded}, \mathsf{alive}, \sim\mathsf{alive}\}$. In the t-de.l.p. (Π_f, Δ) defined by

$$\begin{array}{l} \Pi_\mathsf{f} = \{\langle \sim\mathsf{loaded}, 0\rangle\}, \\ \Delta = \{\delta(3), \delta_0(8), \delta_1(8)\} \cup \{\delta_\ell\}_{\ell \in \{\mathsf{loaded}, \sim\mathsf{loaded}, \mathsf{alive}, \sim\mathsf{alive}\}} \end{array}$$

the literal $\langle \sim\mathsf{alive}, 9\rangle$ is warranted.

5 Logical properties of t-DeLP

After presenting the procedure for computing warrant in t-DeLP, we proceed to show the logical properties of the proposed argumentation framework. These properties, called Rationality Postulates, were proposed by Caminada and Amgoud in [12] (see also [43]) to grant that certain types of counter-intuitive results do not occur in a given argumentation framework.

Definition 11 (Rationality Postulates) The Rationality Postulates, adapted to t-DeLP programs (Π, Δ), read as follows:

Sub-arguments:	if $\mathcal{A}$ is undefeated in $\mathcal{T}_{(\Pi,\Delta)}(\mathcal{A})$, then any sub-argument $\mathcal{A}'$ of $\mathcal{A}$ is also undefeated in $\mathcal{T}_{(\Pi,\Delta)}(\mathcal{A}')$.
Direct Consistency:	$\mathsf{warr}(\Pi, \Delta)$ is consistent.
Indirect Consistency:	$\mathsf{warr}(\Pi, \Delta) \cup \Pi$ is consistent.
Closure:	$\mathrm{Cn}(\mathsf{warr}(\Pi, \Delta) \cup \Pi) \subseteq \mathsf{warr}(\Pi, \Delta)$, i.e. strict consequences of warranted literals are warranted.

These postulates were discussed in [12] for the case of argumentation frameworks defined on top of general defeasible rule-based systems, and using any of the acceptability semantics proposed by Dung for abstract argumentation systems [21]. Given a defeasible rule-based system, these semantics are indeed applied to the abstract argumentation framework $(\mathbf{A}, R)$ generated by such a system: $\mathbf{A}$ being the set of arguments that can be constructed in the rule-based language, and the relation $R(\mathcal{A}, \mathcal{B})$ consisting of a general notion of attack, including rebuts (our notion of attack) and undercuts (not considered in t-DeLP). Then they provide several conditions under which the postulates hold in such sytems. Let us remark that t-DeLP does not exactly fall into the class of systems considered in [12]. Namely, while our arguments easily translate to rule-based systems in [12], our relation of defeat is different from theirs, in particular, our relation of defeat is local, relative to a dialectical tree, and varies across these trees for a given pair of arguments. Moreover, the t-DeLP notion of undefeated argument does not correspond to any of the well-known Dung acceptability

semantics, and in consequence, t-DeLP warrant is also different from its equivalent notion in [12] of *justified conclusions* under any such semantics. See Section 7.1 for a discussion on the relationship between t-DeLP and Dung semantics.

For this reason, we provide in this section direct proofs for the Sub-arguments and Direct Consistency postulates in t-DeLP. Neither Indirect Consistency nor Closure hold for general t-DeLP programs, as it happens in DeLP. However, in the next section we will show that these postulates do hold for a class of programs whose strict rules, roughly speaking, encode mutex constraints.

We first observe that being marked defeated in a dialectical tree can be expressed in the following more convenient form.

Remark 2 An argument $\mathcal{B}$ is marked defeated in an argumentation line $[\mathcal{A}, \ldots, \mathcal{B}] \in \mathcal{T}_{(\Pi,\Delta)}(\mathcal{A})$ iff there is an argument $\mathcal{C}$ marked undefeated in the argumentation line $[\mathcal{A}, \ldots, \mathcal{B}, \mathcal{C}] \in \mathcal{T}_{(\Pi,\Delta)}(\mathcal{A})$.

Lemma 2 *Given a t-d.l.p. (Π, Δ), let $\mathcal{A}_1$ be an argument for $\langle \ell, t\rangle$, and let $\mathcal{A}_2$ be an argument for $\langle \sim\ell, t\rangle$, with $\mathcal{A}_2$ being a defeater for $\mathcal{A}_1$ (at $\mathcal{A}_1$). If $\mathcal{A}_2$ is marked defeated along the argumentation line $[\mathcal{A}_1, \mathcal{A}_2]$ in the dialectical tree $\mathcal{T}_{(\Pi,\Delta)}(\mathcal{A}_1)$, then $\mathcal{A}_2$ is marked defeated in the dialectical tree $\mathcal{T}_{(\Pi,\Delta)}(\mathcal{A}_2)$.*

Proof Let $\mathcal{A}_2$ be a defeater for $\mathcal{A}_1$ at $\mathcal{A}_1$ and assume $\mathcal{A}_2$ is defeated in $\mathcal{T}_{(\Pi,\Delta)}(\mathcal{A}_1)$ and that $\mathcal{A}_2$ is undefeated in $\mathcal{T}_{(\Pi,\Delta)}(\mathcal{A}_2)$. Clearly $[\mathcal{A}_1, \mathcal{A}_2]$ is an argumentation line in $\mathcal{T}_{(\Pi,\Delta)}(\mathcal{A}_1)$, hence there is $\mathcal{A}_3$ such that $[\mathcal{A}_1, \mathcal{A}_2, \mathcal{A}_3]$ is an argumentation line $\mathcal{T}_{(\Pi,\Delta)}(\mathcal{A}_1)$ with $\mathcal{A}_3$ marked undefeated. (See Fig. 8 for an illustration of this proof.)

Now, since $\mathcal{A}_2$ is undefeated in $\mathcal{T}_{(\Pi,\Delta)}(\mathcal{A}_2)$, $[\mathcal{A}_2, \mathcal{A}_3]$ is an argumentation line in $\mathcal{T}_{(\Pi,\Delta)}(\mathcal{A}_2)$ and $\mathcal{A}_3$ is marked defeated. Therefore, there is $\mathcal{A}_4$ such that $[\mathcal{A}_2, \mathcal{A}_3, \mathcal{A}_4]$ is an argumentation line in $\mathcal{T}_{(\Pi,\Delta)}(\mathcal{A}_2)$ with A_4 marked undefeated.

It is easy to check that if some condition (i)-(iii) from Def. 8 fails at the sequence $[\mathcal{A}_1, \mathcal{A}_2, \mathcal{A}_3, \mathcal{A}_4]$, then the same condition already fails either at $[\mathcal{A}_1, \mathcal{A}_2, \mathcal{A}_3]$ or at $[\mathcal{A}_2, \mathcal{A}_3, \mathcal{A}_4]$, contradicting that these two are argumentation line. Thus we have that $[\mathcal{A}_1, \mathcal{A}_2, \mathcal{A}_3, \mathcal{A}_4]$ is an argumentation line in $\mathcal{T}_{(\Pi,\Delta)}(\mathcal{A}_1)$, with $\mathcal{A}_4$ necessarily marked defeated, and hence there must exist $\mathcal{A}_5$ such that $[\mathcal{A}_1, \mathcal{A}_2, \mathcal{A}_3, \mathcal{A}_4, \mathcal{A}_5]$ is an argumentation line in the dialectical tree $\mathcal{T}_{(\Pi,\Delta)}(\mathcal{A}_1)$ with $\mathcal{A}_5$ necessarily marked undefeated.

Iterating this process, one can construct argumentation lines of any finite length $[\mathcal{A}_1, \mathcal{A}_2, \mathcal{A}_3, \ldots, \mathcal{A}_n, \mathcal{A}_{n+1}]$ in $\mathcal{T}_{(\Pi,\Delta)}(\mathcal{A}_1)$, in contradiction with Lemma 1 (2). □

Theorem 1 *Given a t-de.l.p. (Π, Δ), the set of literals $\mathsf{warr}(\Pi, \Delta)$ is consistent.*

Proof Let $\langle \ell, t\rangle \in \mathsf{warr}(\Pi, \Delta)$. Thus, some argument $\mathcal{A}$ for $\langle \ell, t\rangle$ in (Π, Δ) exists that is undefeated in $\mathcal{T}_{(\Pi,\Delta)}(\mathcal{A})$. Let then $\mathcal{B}$ be an arbitrary argument for $\langle \sim\ell, t\rangle$ in (Π, Δ). Assume, towards a contradiction, that $\mathcal{B}$ is undefeated in $\mathcal{T}_{(\Pi,\Delta)}(\mathcal{B})$.

(Case: *$\mathcal{A}$ is a proper defeater for $\mathcal{B}$*) Consider the argumentation line $[\mathcal{B}, \mathcal{A}] \in \mathcal{T}_{(\Pi,\Delta)}(\mathcal{B})$. Since $\mathcal{A}$ is undefeated in $\mathcal{T}_{(\Pi,\Delta)}(\mathcal{A})$, by Lemma 2, we have $\mathcal{A}$ is undefeated in $\mathcal{T}_{(\Pi,\Delta)}(\mathcal{B})$. Hence $\mathcal{B}$ is marked defeated in $\mathcal{T}_{(\Pi,\Delta)}(\mathcal{B})$. Since $\mathcal{B}$ was arbitrary, $\langle \sim\ell, t\rangle \notin \mathsf{warr}(\Pi, \Delta)$.

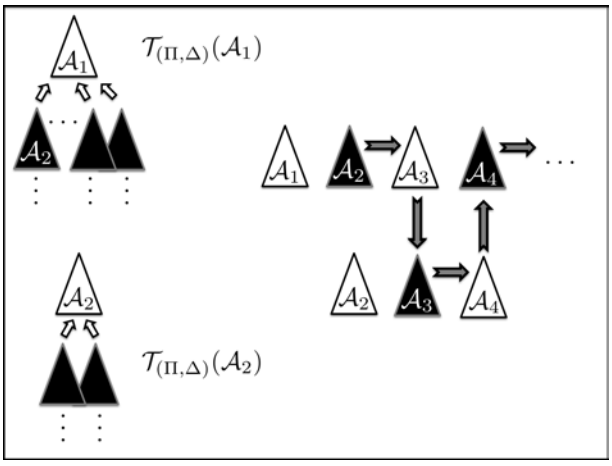

Fig. 8 Constructing an infinite sequence of increasing argumentation lines: white and black triangles represent, respectively, undefeated and defeated arguments in the corresponding dialectical trees.

(Case: *$\mathcal{A}$ is not a proper defeater for $\mathcal{B}$*). By Proposition 2 (ii), $\mathcal{B}$ is a defeater for $\mathcal{A}$, so $[\mathcal{A}, \mathcal{B}]$ is an arg. line in $\mathcal{T}_{(\Pi,\Delta)}(\mathcal{A})$. Since $\mathcal{A}$ is undefeated in this tree, $\mathcal{B}$ must be defeated in this tree. Then again by Lemma 2, we have $\mathcal{B}$ is also defeated in $\mathcal{T}_{(\Pi,\Delta)}(\mathcal{B})$. Since $\mathcal{B}$ was arbitrary, $\langle \sim\ell, t\rangle \notin \mathsf{warr}(\Pi, \Delta)$. □

The previous results, with appropriate modifications, remain valid for other argumentative frameworks in the DeLP family, like ODeLP [13]. Theorem 1 can be seen as a generalization of their Direct Consistency result when de.l.p.s are allowed strict rules. Similar results to those of Lemma 2 and Theorem 1 are made for the case of DeLP in [50, Props 1 and 2]. Also, the later results in Section 6 are in line with [12] for defeasible logics. However, t-DeLP does not seem to reduce to the logical frameworks considered in these two contributions.

In Lemma 2, we showed the property of *being defeated* for $\mathcal{A}_2$ is downward preserved from $[\mathcal{A}_1, \mathcal{A}_2]$ to $[\mathcal{A}_2]$. (Or, conversely, *being undefeated* is upward preserved from $[\mathcal{A}_2]$ to any existing line of the form $[\mathcal{A}_1, \mathcal{A}_2]$.) This downward property can be generalized to *defeated* arguments in arbitrary interfering positions. Another upward preservation result holds for *undefeated* arguments in supporting positions.

Corollary 1 *Let $\Lambda = [\mathcal{A}_1, \mathcal{A}_2, \ldots, \mathcal{A}]$ be an argumentation line in $\mathcal{T}_{(\Pi,\Delta)}(\mathcal{A}_1)$. Then*

(1) if $\mathcal{A} = \mathcal{A}_{2n+1}$ is undefeated in Λ, then in the corresponding arg. line $[\mathcal{A}_2, \ldots, \mathcal{A}]$ the (now interfering) argument $\mathcal{A}$ is undefeated;
(2) if $\mathcal{A} = \mathcal{A}_{2n}$ is defeated in Λ, then in the corresponding arg. line $[\mathcal{A}_2, \ldots, \mathcal{A}]$ the (now supporting) $\mathcal{A}$ is defeated.

Finally, we address the postulate of Sub-arguments for general t-DeLP programs.

Lemma 3 *If $\Lambda = [\mathcal{A}_1, \ldots, \mathcal{A}_n]$ is an arg. line for $\mathcal{A}_1$ and $\mathcal{A}'_1 \supseteq \mathcal{A}_1(\sim\mathsf{concl}(\mathcal{A}_2))$, then*

$$\Lambda' = [\mathcal{A}'_1, \ldots, \mathcal{A}_n] \text{ is an argumentation line}$$

Proof Note first that $\mathcal{A}_2$ is a defeater for $\mathcal{A}'_1$. It remains to be checked that $\Lambda' = [\mathcal{A}'_1, \ldots, \mathcal{A}_n]$ satisfies (i)-(iii) from Def. 8. Condition (i) is obvious: if a contradiction exists from the supporting elements of Λ', then, since their union is included in the union of supporting elements in Λ, the same contradiction would occur in Λ; similarly for interfering arguments. Condition (ii). If $\mathcal{A}_{2k+1} = \mathcal{A}'_1(\sim\mathsf{concl}(\mathcal{A}_2))$, then the fact $\mathcal{A}'_1(\sim\mathsf{concl}(\mathcal{A}_2)) = \mathcal{A}_1(\sim\mathsf{concl}(\mathcal{A}_2))$ implies the same violation of (ii) would occur in Λ; the remaining cases are direct. Condition (iii). The interesting case are the triples $[\mathcal{A}_1, \mathcal{A}_2, \mathcal{A}_3]$ and $[\mathcal{A}'_1, \mathcal{A}_2, \mathcal{A}_3]$. But being a blocking or proper defeater only depends on the attacked sub-argument. Thus, if $\mathcal{A}_3$ is blocking for $\mathcal{A}_2$ and this is blocking for $\mathcal{A}'_1$, then the same occurs for the other triple $[\mathcal{A}_1, \mathcal{A}_2, \mathcal{A}_3]$, contradicting that Λ is an arg. line. □

Note that this Lemma implies the validity of the Sub-arguments postulate for arbitrary t-de.l.p. programs (Π, Δ).

Corollary 2 *Given a t-de.l.p.* (Π, Δ)*, if* $\mathcal{A}_1$ *is undefeated in* $\mathcal{T}_{(\Pi,\Delta)}(\mathcal{A}_1)$ *and* $\mathcal{A}'_1 \subseteq \mathcal{A}_1$ *is an argument, then* $\mathcal{A}'_1$ *is undefeated in* $\mathcal{T}_{(\Pi,\Delta)}(\mathcal{A}'_1)$.

Proof Using Lemma 3, we have that if $\mathcal{A}'_1 \subseteq \mathcal{A}_1$ is an argument, then the dialectical tree $\mathcal{T}_{(\Pi,\Delta)}(\mathcal{A}'_1)$ is a sub-tree of $\mathcal{T}_{(\Pi,\Delta)}(\mathcal{A}_1)$, containing those maximal argumentation lines $[\mathcal{A}'_1, \mathcal{A}_2, \ldots]$ with $\sim\mathsf{concl}(\mathcal{A}_2) \in \mathsf{literals}[\mathcal{A}'_1]$. But, since $\mathcal{A}_2$ is defeated in $\mathcal{T}_{(\Pi,\Delta)}(\mathcal{A}_1)$ so is $\mathcal{A}_2$ in $\mathcal{T}_{(\Pi,\Delta)}(\mathcal{A}'_1)$. Thus, $\mathcal{A}'_1$ is undefeated in $\mathcal{T}_{(\Pi,\Delta)}(\mathcal{A}'_1)$. □.

6 t-DeLP with mutex constraints

In the previous section, we have seen in Example 4 (2) and Example 5 programs (Π, Δ) whose set of strict rules $\Pi_{\mathsf{r}} \subseteq \Pi$ exclusively consists of mutex rules of the form $\langle\sim q, t\rangle \leftarrow \langle p, t\rangle$. Although in those particular examples the expected outputs $\mathsf{warr}(\Pi, \Delta)$ are properly captured by the procedure for warrant, the Indirect Consistency and Closure postulates do not seem provable in the general case of t-DeLP programs with mutex rules. However, in this section we show that by slightly modifying the procedure for warrant we are able to guarantee the fulfillment of these postulates. The modification essentially consists in strengthening the notion of consistency: instead of using a strict rule $\langle\sim q, t\rangle \leftarrow \langle p, t\rangle$ to detect a conflict (in the usual way) between arguments for literals p and q in a mutex set, we extend the notion of inconsistency (attacks, etc) to hold among pair of mutex literals. Recall from Section 3 that the intended representation of logical constraints is by a family $\mathbf{M}$ of mutex sets. A mutex set X is a set of positive literals $X = \{p, q, r, \ldots\} \subseteq \mathsf{Var}$ expressing that they are pairwise incompatible. Hence each mutex set X induces a set of non-durative strict rules $\Pi_X = \{\langle\sim q, t\rangle \leftarrow \langle p, t\rangle \mid p, q \in X, t \in \mathbb{N}\}$. Given a family of mutex sets $\mathbf{M}$ we will denote by $\Pi_{\mathbf{M}}$ the union of the sets of strict rules Π_X for each $X \in \mathbf{M}$.

Given a mutex family $\mathbf{M}$, one can naturally strengthen the notion of consistency for a set of literals by taking into account the mutex constraints.

Definition 12 ($\mathbf{M}$-consistency) Given a mutex family $\mathbf{M}$ and a set of literals Γ, we say that Γ is $\mathbf{M}$-*consistent* iff Γ is consistent and moreover there are no $\langle p, t\rangle, \langle q, t\rangle \in$

Γ and $X \in \mathbf{M}$ such that $p, q \in X$. For a set of literals Γ and rules Π, we say $\Gamma \cup \Pi$ is $\mathbf{M}$-consistent iff the set of literals $\mathrm{Cn}(\Gamma \cup \Pi)$ is $\mathbf{M}$-consistent.

In what follows, we extend a t-DeLP program (Π, Δ) together with a family of mutex constraints $\mathbf{M} = \{X_1, X_2, \dots\}$. The notion of $\mathbf{M}$-consistency refines the notion of argument, attack and defeat and spreads to other related definitions like argumentation line, dialectical tree and warrant from Section 4.

Definition 13 ($\mathbf{M}$-argument, $\mathbf{M}$-attack, $\mathbf{M}$-defeat) Let (Π, Δ) be a t-de.l.p. and $\mathbf{M}$ a mutex family. An $\mathbf{M}$-*argument* in (Π, Δ) is defined as in Definition 4 replacing condition (2) by

(2') $\Pi \cup \mathcal{A}_\Delta$ is $\mathbf{M}$-consistent

Let $\mathcal{A}_1, \mathcal{A}_0$ be $\mathbf{M}$-arguments in (Π, Δ), with $\mathsf{concl}(\mathcal{A}_1) = \langle \ell, t \rangle$. Then:

- We say $\mathcal{A}_1$ $\mathbf{M}$-*attacks* $\mathcal{A}_0$ if there exists $\langle \ell', t \rangle \in \mathsf{literals}[\mathcal{A}_0]$ such that either $\ell' = \sim\ell$ or $\{\ell, \ell'\} \subseteq X$ for some $X \in \mathbf{M}$. In this case we say $\mathcal{A}_1$ $\mathbf{M}$-attacks $\mathcal{A}_0$ at the sub-argument $\mathcal{A}_0(\langle \ell', t \rangle)$.
- Let $\mathcal{A}_1$ $\mathbf{M}$-attack $\mathcal{A}_0$ at the sub-argument $\mathcal{A}_0(\langle \ell', t \rangle)$. We say $\mathcal{A}_1$ is a *proper* $\mathbf{M}$-*defeater for* $\mathcal{A}_0$, denoted $\mathcal{A}_1 \succ \mathcal{A}_0$, iff
  - $\mathsf{base}(\mathcal{A}_1) \supsetneq \mathsf{base}(\mathcal{A}_0)$, or
  - there is $t' < t$ such that $\mathcal{A}_0 = \mathcal{A}_1(\langle \ell', t' \rangle) \cup \{\delta_{\ell'}(t'')\}_{t' \leq t'' < t}$

  If $\mathcal{A}_1 \not\succ \mathcal{A}_0$ and $\mathcal{A}_0 \not\succ \mathcal{A}_1$ then we say that $\mathcal{A}_1$ are $\mathcal{A}_0$ are *blocking* $\mathbf{M}$-*defeaters*.

From there, the notions of $\mathbf{M}$-argumentation line, $\mathbf{M}$-dialectical tree $\mathcal{T}^{\mathbf{M}}_{(\Pi,\Delta)}(\cdot)$ and the corresponding notion of warrant $\mathsf{warr}^{\mathbf{M}}(\Pi, \Delta)$ are defined in the natural way from $\mathcal{T}_{(\Pi_\mathsf{f},\Delta)}(\cdot)$ and $\mathsf{warr}(\Pi_\mathsf{f}, \Delta)$ using now the latter notions of $\mathbf{M}$-argument, $\mathbf{M}$-attack and $\mathbf{M}$-defeater. It is routine to check that all of the previous results remain valid for $\mathsf{warr}^{\mathbf{M}}(\Pi, \Delta)$.

As a consequence, we have the next result about Direct Consistency.

Corollary 3 *Let (Π, Δ) be a t-de.l.p. and $\mathbf{M}$ a mutex family such that Π is $\mathbf{M}$-consistent. Then, $\mathsf{warr}^{\mathbf{M}}(\Pi, \Delta)$ is $\mathbf{M}$-consistent.*

Next we define the output of a t-DeLP program with an associated mutex family $\mathbf{M}$ as the closure of $\mathsf{warr}^{\mathbf{M}}(\Pi, \Delta)$ with the set of strict rules $\Pi_{\mathbf{M}}$ induced by $\mathbf{M}$.

Definition 14 ($\mathbf{M}$-output) Let (Π, Δ) be a t-de.l.p. with an associated mutex family $\mathbf{M}$, such that $\Pi \cup \Pi_{\mathbf{M}}$ is consistent. Then we define:

$$\mathsf{Output}^{\mathbf{M}}(\Pi, \Delta) = \mathrm{Cn}(\mathsf{warr}^{\mathbf{M}}(\Pi, \Delta) \cup \Pi_{\mathbf{M}}).$$

Given the previous definitions, we can prove some results for the Rationality postulates for the outputs of t-DeLP programs (Π, Δ) with an associated mutex familty $\mathbf{M}$ in the particular case when Π only contains strict facts, i.e. when $\Pi = \Pi_\mathsf{f}$, or equivalently when $\Pi_\mathsf{r} = \emptyset$. Note that it does not make actually sense to ask whether the Sub-arguments postulate holds for $\mathsf{Output}^{\mathbf{M}}(\Pi, \Delta)$, since besides the arguments from (Π, Δ), this notion relies on the logical closure with $\Pi_{\mathbf{M}}$. On the other hand, it is easy to show that any derivation of a literal in $\mathsf{Output}^{\mathbf{M}}(\Pi, \Delta)$ still corresponds to the conclusion of an *extended* argument, namely, an argument in (Π, Δ) extended with at most a rule in $\Pi_{\mathbf{M}}$.

Lemma 4 *Given a set of literals Γ, for any $\langle \ell, t\rangle \in \mathrm{Cn}(\Gamma \cup \Pi_{\mathbf{M}})$, we have $\langle \ell, t\rangle \in \mathrm{Cn}(\Gamma \cup \{\delta\})$, for some $\delta \in \Pi_{\mathbf{M}}$.*

Proof Let $A \subseteq \Gamma \cup \Pi_{\mathbf{M}}$ be such that $A \vdash \langle \ell, t\rangle$. In case $\langle \ell, t\rangle \in \Gamma$, just select any rule from $\Pi_{\mathbf{M}}$. Otherwise, if $\langle \ell, t\rangle \notin \Gamma$, then there must exist a rule $\delta \in \Pi_{\mathbf{M}}$ such that $\mathsf{head}(\delta) = \langle \sim q, t\rangle$ and $\langle \ell, t\rangle = \langle \sim q, t\rangle$. Then, since $\mathsf{body}(\delta)$ is a set of positive atoms, $\mathsf{head}(\delta) \cap \mathsf{body}[\Pi_{\mathbf{M}}] = \emptyset$, so it must be $\mathsf{body}(\delta) \subseteq \Gamma$. Thus, $\langle \sim q, t\rangle = \langle \ell, t\rangle \in \mathrm{Cn}(\Gamma \cup \{\delta\})$. □

Theorem 2 *Let consider the class of t-DeLP programs (Π, Δ) with an associated mutex family $\mathbf{M}$, such that $\Pi_r = \emptyset$ and Π is $\mathbf{M}$-consistent. Then Direct Consistency, Indirect Consistency and Closure are satisfied by the $\mathsf{Output}^{\mathbf{M}}$ inference procedure.*

Proof We show the three properties for any t-DeLP program (Π, Δ) with an associated mutex family $\mathbf{M}$, such that $\Pi_r = \emptyset$ and Π is $\mathbf{M}$-consistent.

(i) *Direct Consistency.* Assume the contrary, so there exists a pair $\langle p, t\rangle, \langle \sim p, t\rangle$ of contradictory elements in $\mathsf{Output}^{\mathbf{M}}(\Pi, \Delta) = \mathrm{Cn}(\mathsf{warr}^{\mathbf{M}}(\Pi, \Delta) \cup \Pi_{\mathbf{M}})$. Consider the cases:
- Case $\langle p, t\rangle, \langle \sim p, t\rangle \in \mathsf{warr}^{\mathbf{M}}(\Pi, \Delta)$: impossible, since $\mathsf{warr}^{\mathbf{M}}(\Pi, \Delta)$ is consistent by Corollary 3.
- Case $\langle p, t\rangle \in \mathsf{warr}^{\mathbf{M}}(\Pi, \Delta)$, $\langle \sim p, t\rangle \notin \mathsf{warr}^{\mathbf{M}}(\Pi, \Delta)$. In this case we have $\langle \sim p, t\rangle \in \mathrm{Cn}(\mathsf{warr}^{\mathbf{M}}(\Pi, \Delta) \cup \Pi_{\mathbf{M}})$. Applying Lemma 4 with $\Gamma = \mathsf{warr}^{\mathbf{M}}(\Pi, \Delta)$, let $\delta \in \Pi_{\mathbf{M}}$ be such that $\langle \sim p, t\rangle \in \mathrm{Cn}(\mathsf{warr}^{\mathbf{M}}(\Pi, \Delta) \cup \{\delta\})$. Thus, $\mathsf{body}(\delta) = \{\langle q, t\rangle\} \subseteq \mathsf{warr}^{\mathbf{M}}(\Pi, \Delta)$, for some $p, q \in X \in \mathbf{M}$. But this and the initial assumption $\langle p, t\rangle \in \mathsf{warr}^{\mathbf{M}}(\Pi, \Delta)$, contradict the fact that $\mathsf{warr}^{\mathbf{M}}(\Pi, \Delta)$ is $\mathbf{M}$-consistent.
- Case $\langle p, t\rangle \notin \mathsf{warr}^{\mathbf{M}}(\Pi_{\mathsf{f}}, \Delta)$. By Lemma 4, this is impossible: $\mathsf{head}[\Pi_{\mathbf{M}}]$ is a set of negated atoms, so $\langle p, t\rangle \notin \mathsf{warr}^{\mathbf{M}}(\Pi, \Delta)$ would imply $\langle p, t\rangle \notin \mathsf{warr}^{\mathbf{M}}(\Pi, \Delta)$.

Thus, in either case we reach a contradiction.

(ii) *Closure*. Let $\langle \ell, t\rangle \in \mathrm{Cn}(\mathsf{Output}^{\mathbf{M}}(\Pi, \Delta) \cup \Pi)$. We show $\langle \ell, t\rangle \in \mathsf{Output}^{\mathbf{M}}(\Pi, \Delta)$. The cases $\langle \ell, t\rangle \in \Pi_{\mathsf{f}}$ and $\langle \ell, t\rangle \in \mathsf{Output}^{\mathbf{M}}(\Pi, \Delta)$ are immediate, so we can assume that $\langle \ell, t\rangle \notin \mathsf{Output}^{\mathbf{M}}(\Pi, \Delta)$. By Lemma 4 with $\Gamma = \mathsf{Output}^{\mathbf{M}}(\Pi, \Delta)$, let $\delta \in \Pi_{\mathbf{M}}$ be such that $\langle \ell, t\rangle \in \mathrm{Cn}(\mathsf{Output}^{\mathbf{M}}(\Pi_{\mathsf{f}}, \Delta) \cup \{\delta\})$. Thus, $\mathsf{body}(\delta) \in \mathsf{Output}^{\mathbf{M}}(\Pi, \Delta)$, but since $\mathsf{body}(\delta)$ consists of a positive atom, and hence not an atom in $\mathsf{head}[\Pi_{\mathbf{M}}]$, we must have $\mathsf{body}(\delta) \subseteq \mathsf{warr}^{\mathbf{M}}(\Pi_{\mathsf{f}}, \Delta)$. Thus,
$$\langle \ell, t\rangle \in \mathrm{Cn}(\mathsf{warr}^{\mathbf{M}}(\Pi_{\mathsf{f}}, \Delta) \cup \Pi_{\mathbf{M}}) = \mathsf{Output}^{\mathbf{M}}(\Pi, \Delta).$$

(iii) *Indirect Consistency*. This follows from the previous results on closure and direct consistency for $\mathsf{Output}^{\mathbf{M}}(\Pi, \Delta)$. □

Finally, let us compare this proposal $\mathsf{Output}^{\mathbf{M}}(\cdot, \cdot)$ with a more straightforward approach based on t-DeLP warrant alone, where the program directly contains mutex strict rules $\Pi \cup \Pi_{\mathbf{M}}$. [7] The two approaches are compared to each other with the help of an example, showing that the latter approach $\mathsf{warr}(\Pi \cup \Pi_{\mathbf{M}}, \Delta)$ might not satisfy indirect consistency or closure w.r.t. these mutex rules $\Pi_{\mathbf{M}}$.

[7] We are thankful to a reviewer for pointing out this question.

Example 8 Let (Π, Δ) and $\mathbf{M}, \Pi_{\mathbf{M}}$ be defined by

$$
\begin{array}{rcl}
\Pi & = & \{\langle r, 0\rangle\} \\
\Delta & = & \left\{\begin{array}{c} \langle \sim p, 1\rangle \prec \langle r, 0\rangle, \\ \langle q, 1\rangle \prec \langle r, 0\rangle, \\ \langle p, t+100\rangle \prec \langle \sim p, t\rangle, \\ \langle q, t+1\rangle \prec \langle q, t\rangle \end{array}\right\} \begin{array}{l} = \delta_0 \\ = \delta_1 \\ = \delta_2(t) \\ = \delta_q(t) \end{array} \\
\mathbf{M} & = & \{\, \{p, q\} \,\} \\
\Pi_{\mathbf{M}} & = & \left\{\begin{array}{c} \langle \sim p, t\rangle \leftarrow \langle q, t\rangle, \\ \langle \sim q, t\rangle \leftarrow \langle p, t\rangle \end{array}\right\} \begin{array}{l} = \delta_p^{\mathbf{M}}(t) \\ = \delta_q^{\mathbf{M}}(t) \end{array}
\end{array}
$$

Then consider the next arguments. Those on the right are only available in the program $(\Pi \cup \Pi_{\mathbf{M}}, \Delta)$.

both in (Π, Δ) and $(\Pi \cup \Pi_{\mathbf{M}}, \Delta)$	only in $(\Pi \cup \Pi_{\mathbf{M}}, \Delta)$
$\mathcal{A} = \{\langle r, 0\rangle, \delta_0, \delta_2(1)\}$	$\mathcal{A}^+ = \mathcal{A} \cup \{\delta_p^{\mathbf{M}}(101)\}$
$\mathsf{concl}(\mathcal{A}) = \langle p, 101\rangle$	$\mathsf{concl}(\mathcal{A}^+) = \langle \sim q, 101\rangle$
$\mathcal{B} = \{\langle r, 0\rangle, \delta_1\} \cup \{\delta_q(t)\}_{1 \le t \le 100}$	$\mathcal{B}^+ = \mathcal{B} \cup \{\delta_q^{\mathbf{M}}(101)\}$
$\mathsf{concl}(\mathcal{B}) = \langle q, 101\rangle$	$\mathsf{concl}(\mathcal{B}^+) = \langle \sim p, 101\rangle$

The direct approach $\mathsf{warr}(\Pi \cup \Pi_{\mathbf{M}}, \Delta)$, on the one hand makes the arguments $\mathcal{A}^+, \mathcal{B}^+$ available. Observe that while $\mathcal{A}^+$ defeats $\mathcal{B}$ as expected, because the presence of persistence rules $\{\delta_q^{\mathbf{M}}(101)\}$ in the argument $\mathcal{B}$ makes it weaker than argument $\mathcal{A}^+$, so we have $\mathcal{A}^+ \succ \mathcal{B}$. However this does not apply to the other pair of arguments $\mathcal{A}$ and $\mathcal{B}^+$, so we have $\mathcal{A} \prec\!\succ \mathcal{B}^+$. For the same reason, $\mathcal{A}^+ \prec\!\succ \mathcal{B}^+$. Hence, the dialectical trees become

$$
\begin{array}{lcll}
\mathcal{T}_{(\Pi \cup \Pi_{\mathbf{M}}, \Delta)}(\mathcal{A}) & = & \{[\mathcal{A}, \mathcal{B}^+, \mathcal{A}^+]\} & \\
\mathcal{T}_{(\Pi \cup \Pi_{\mathbf{M}}, \Delta)}(\mathcal{A}^+) & = & \{[\mathcal{A}^+, \mathcal{B}^+]\} & \text{adding } [\mathcal{A}] \text{ is against (ii)} \\
\mathcal{T}_{(\Pi \cup \Pi_{\mathbf{M}}, \Delta)}(\mathcal{B}) & = & \{[\mathcal{B}, \mathcal{A}^+, \mathcal{B}^+]\} & \text{idem} \\
\mathcal{T}_{(\Pi \cup \Pi_{\mathbf{M}}, \Delta)}(\mathcal{B}^+) & = & \{[\mathcal{B}^+, \mathcal{A}^+]\} &
\end{array}
$$

so we conclude $\langle p, 101\rangle, \langle q, 101\rangle \in \mathsf{warr}(\Pi \cup \Pi_{\mathbf{M}}, \Delta)$ while $\langle \sim q, 101\rangle, \langle \sim p, 101\rangle \notin \mathsf{warr}(\Pi \cup \Pi_{\mathbf{M}}, \Delta)$; in other words, the strict rules $\Pi_{\mathbf{M}}$ are not enforced, and the set of warranted literals is not $\mathbf{M}$-consistent, hence not consistent with $\Pi_{\mathbf{M}}$.

On the other hand, we have in (Π, Δ) that $\mathcal{A}$ and $\mathcal{B}$ $\mathbf{M}$-attack each other, and moreover $\mathcal{A}$ is a proper $\mathbf{M}$-defeater for $\mathcal{B}$, so $\mathcal{A} \succ \mathcal{B}$. An easy computation shows that $\mathcal{T}_{(\Pi, \Delta)}(\mathcal{A}) = [\mathcal{A}]$ and $\mathcal{T}_{(\Pi, \Delta)}(\mathcal{B}) = [\mathcal{B}, \mathcal{A}]$. As a result, $\langle p, 101\rangle \in \mathsf{warr}^{\mathbf{M}}(\Pi, \Delta)$ and $\langle q, 101\rangle \notin \mathsf{warr}^{\mathbf{M}}(\Pi, \Delta)$. In the last step to compute $\mathsf{Output}^{\mathbf{M}}(\cdot, \cdot)$, we obtain $\langle p, 2\rangle, \langle \sim q, 2\rangle \in \mathrm{Cn}(\mathsf{warr}^{\mathbf{M}}(\Pi, \Delta) \cup \Pi_{\mathbf{M}}) = \mathsf{Output}^{\mathbf{M}}(\Pi, \Delta)$.

7 Some further remarks on the comparison of t-DeLP with Dung semantics, DeLP and the TDR system

In this section, we first discuss the relationship between the dialectical tree based-semantics of t-DeLP and Dung acceptability semantics for abstract argumentation frameworks. Then we report on the comparison of some particular aspects of t-DeLP with DeLP [23] and also with another temporal extension of DeLP in the literature called Temporal Defeasible Reasoning (TDR) [5].

7.1 t-DeLP and Dung acceptability semantics

Let us briefly review the semantics from [21] proposed for *abstract argumentation frameworks*. The latter simply consists of a relation R, called *attack*, on a set of (unstructured) elements $\mathbf{A} = \{\mathcal{A}, \ldots\}$, called *arguments*, i.e. a pair $(\mathbf{A}, R)$. The so-called acceptability semantics try to capture different intuitions about which subsets $\mathbb{E} \subseteq \mathbf{A}$ are collectively acceptable (called *extensions*), given the attack relation. For example,

$$\mathbb{E} \text{ is } \textit{conflict-free} \text{ iff no } \mathcal{A}, \mathcal{B} \in \mathbb{E} \text{ exist with } R(\mathcal{A}, \mathcal{B}).$$

Other intuitive conditions upon extensions are defined from the notion of defense: a subset $\mathbb{E}$ *defends* $\mathcal{A}$ iff

$$\text{for each } \mathcal{B} \text{ attacking } \mathcal{A}, \text{ there exists } \mathcal{C} \in \mathbb{E} \text{ that attacks } \mathcal{B}.$$

By denoting $\mathcal{F}(\mathbb{E}) = \{\mathcal{B} \mid \mathbb{E} \text{ defends } \mathcal{B}\}$, we say a conflict-free extension $\mathbb{E}$ is

admissible	iff	$\mathbb{E}$ defends its arguments ($\mathbb{E} \subseteq \mathcal{F}(\mathbb{E})$)
complete	iff	$\mathcal{F}(\mathbb{E}) = \mathbb{E}$
grounded	iff	$\mathbb{E}$ is the $\subseteq$-minimal complete extension
preferred	iff	$\mathbb{E}$ is a $\subseteq$-maximal complete extension
stable	iff	$\mathbb{E}$ is a preferred extension attacking each argument in $\mathbf{A} \setminus \mathbb{E}$

For each semantics $\mathcal{X} = \{\text{admissible}, \ldots\}$, the (skeptical) *justified or acceptable arguments* according to $\mathcal{X}$ are defined as the set of arguments in the intersection of all the $\mathcal{X}$-extensions: $\bigcap\{\mathbb{E} \mid \mathbb{E} \text{ is an } \mathcal{X}-\text{extension}\}$.

If we directly rephrase the acceptability semantics from [21] and the related definitions above, there is still a mismatch between Dung's acceptability and acceptability in t-DeLP (i.e. undefeated arguments). To see this, first note that the abstract notion of attack R corresponds to our notion of defeat *relative to* some dialectical tree $\mathcal{T}_{(\Pi,\Delta)}(\mathcal{A})$. Correspondingly, the (relevant) defense of an argument $\mathcal{A}$ will take place only in its own dialectical tree $\mathcal{T}_{(\Pi,\Delta)}(\mathcal{A})$, so the $\mathcal{F}$ function would be expressed by

$$\mathcal{F}(\mathbb{E}) = \{\mathcal{A} \mid \forall \mathcal{B} \in \mathbf{A}\ \exists \mathcal{C} \in \mathbb{E}\ ([\mathcal{A}, \mathcal{B}] \in \mathcal{T}_{(\Pi,\Delta)}(\mathcal{A}) \ \Rightarrow\ [\mathcal{A}, \mathcal{B}, \mathcal{C}] \in \mathcal{T}_{(\Pi,\Delta)}(\mathcal{A})\}^{8}$$

Also, note that in a t-de.l.p. (Π, Δ) there is a unique notion of *extension*, or set of "acceptable" arguments; namely, those arguments $\mathcal{A}$ that are undefeated in $\mathcal{T}_{(\Pi,\Delta)}(\mathcal{A})$.

[8] Stronger definitions of the defense of an argument $\mathcal{A}$ and of the $\mathcal{F}$ function exist; e.g. where $\mathcal{A}$ is "defended" in all its occurrences in dialectical trees of arbitrary arguments $\mathcal{T}_{(\Pi,\Delta)}(\mathcal{D})$, not just in its own tree $\mathcal{T}_{(\Pi,\Delta)}(\mathcal{A})$. But these are also prey to the counter-example below.

That the t-DeLP procedure for undefeated arguments can define a non-admissible extension $\mathbb{E} \nsubseteq \mathcal{F}(\mathbb{E})$ is shown next.

Example 9 Let $\mathcal{A}, \mathcal{B}, \mathcal{C}$ be the arguments for, resp., $\langle s, 1\rangle, \langle \sim s, 1\rangle$ and $\langle s, 1\rangle$ consisting of: (1) a single rule each $\langle s, 1\rangle \prec \langle p, 0\rangle$ and $\langle \sim s, 1\rangle \prec \langle p, 0\rangle, \langle q, 0\rangle$ and $\langle s, 1\rangle \prec \langle r, 0\rangle$, resp., and (2) the facts given by the body of the corresponding rule. Moreover, let (Π, Δ) be the program defined just by these strict facts and defeasible rules from (1) and (2). Then,

$$\mathcal{T}_{(\Pi,\Delta)}(\mathcal{A}) = \{[\mathcal{A}, \mathcal{B}, \mathcal{C}]\} \quad \mathcal{T}_{(\Pi,\Delta)}(\mathcal{B}) = \{[\mathcal{B}, \mathcal{C}]\} \quad \mathcal{T}_{(\Pi,\Delta)}(\mathcal{C}) = \{[\mathcal{C}, \mathcal{B}]\}$$

so $\mathcal{A}$ is undefeated and $\mathcal{B}, \mathcal{C}$ are defeated; but there is only $\mathcal{C}$ to defend $\mathcal{A}$ (from $\mathcal{B}$). As a result, $\mathbb{E} = \{\mathcal{A}\} \nsubseteq \emptyset = \mathcal{F}(\mathbb{E})$, and thus extension $\mathbb{E}$ is not admissible.

Since complete extensions are defined by strengthening the condition for admissible extensions, namely $\mathbb{E} = \mathcal{F}(\mathbb{E})$, this counter-example also shows that the t-DeLP extensions do not correspond to the semantics based on (subsets of) the complete extensions. This includes the remaining four semantics: complete, grounded, preferred and stable.

7.2 Defeat criteria in DeLP and t-DeLP.

Both DeLP and t-DeLP are defeasible argumentation-based logic programming frameworks. The former is defined by criteria of defeasibility expressing a preference for arguments with more direct inference steps. This is the reason to prefer {*penguins do not fly*} over {*penguins are birds*, *birds fly*}. In [23], the authors use the so-called *generalized specificity* to formalize this idea of defeat as described next. (Recall, arguments in [23] are of the form $\langle \mathcal{A}, \ell\rangle$ for some conclusion ℓ. Moreover, in this work arguments are identified only in terms of the defeasible information they make use of, while abstracting from strict information.)

Definition 15 (DeLP-specificity) Let (Π, Δ) be a de.l.p., and let Π_{r} be the set of all strict rules from Π (without including facts.) Let $\mathcal{F}$ be the set of all literals that are derivable from (Π, Δ). Let $\langle \mathcal{A}_1, \ell_1\rangle$ and $\langle \mathcal{A}_2, \ell_2\rangle$ be two arguments obtained from (Π, Δ). $\langle \mathcal{A}_1, \ell_1\rangle$ is *strictly more specific* than $\langle \mathcal{A}_2, \ell_2\rangle$ if the following conditions hold:

1. for all $H \subseteq \mathcal{F}$: if $\Pi_{\mathsf{r}} \cup H \cup \mathcal{A}_1 \vdash \ell_1$ and $\Pi_{\mathsf{r}} \cup H \nvdash \ell_1$, then $\Pi_{\mathsf{r}} \cup H \cup \mathcal{A}_2 \vdash \ell_2$, and
2. there exists $H' \subseteq \mathcal{F}$ such that: $\Pi_{\mathsf{r}} \cup H' \cup \mathcal{A}_2 \vdash \ell_2$ and $\Pi_{\mathsf{r}} \cup H' \nvdash \ell_2$ and $\Pi_{\mathsf{r}} \cup H' \cup \mathcal{A}_1 \nvdash \ell_1$.

When the conditions for 1 are met, i.e. $\Pi_{\mathsf{r}} \cup H \cup \mathcal{A} \vdash \ell$ and $\Pi_{\mathsf{r}} \cup H \nvdash \ell_1$, we say H is an *activation set* for $\langle \mathcal{A}, \ell\rangle$. The idea of DeLP-specificity is to prefer arguments with fewer activation sets (in the sense of inclusion). In other words, to prefer the existence of less combinations of intermediate steps sufficing for the conclusion.

Turning back to t-DeLP, when activation sets are instead defined from strict facts in Π_{f}, this specificity criterion in Definition 7 turns into our first criterion for t-DeLP-preference based on a comparison between $\mathsf{base}(\mathcal{A}_1)$ and $\mathsf{base}(\mathcal{A}_2)$. Notice this maneuver is possible only if we fix the strict information in the arguments. On the other hand, the second criterion in Def. 7, based on the use of persistence rules, is also inspired by the notion of activation set, but with a preference for *more activation sets*. In this sense we have the following equivalence:

$\mathcal{A}_1$ is longer than $\mathcal{A}_2$ iff there is a subset $\Delta'_{\mathsf{p}} \subseteq \Delta_{\mathsf{p}}$ such that in $(\Pi, \Delta \setminus \Delta'_{\mathsf{p}})$ the activation sets for $\mathcal{A}_1$ strictly contain those for $\mathcal{A}_2$.

7.3 t-DeLP and the framework of Temporal Defeasible Reasoning (TDR)

In the TDR framework [5], literals (hence conclusions) of arguments are primitively associated with both discrete intervals and time-points. For instance, (using our own notation) $\mathsf{head}(\delta) = \langle \ell, [t+1, t+3] \rangle$ expresses that if $\delta \in \Delta$ applies we defeasibly conclude that ℓ holds from $t+1$ to $t+3$. In TDR, conflicts between two interval-valued arguments, e.g. $(\mathcal{A}, \langle \ell, [t+1, t+4] \rangle)$ and $(\mathcal{B}, \langle \sim\ell, [t+2, t+5] \rangle)$ attacking each other do so at the intersection of these intervals $[t+2, t+4]$.

Another significant difference between TDR and t-DeLP lies again in the defeat criteria. In particular, when persistence rules are involved in the comparison between two contending arguments. In TDR, an argument that contains persistence rules is less preferred than an argument which does not. In contrast, in t-DeLP (Def. 7 above) the comparison is made locally (at the level of sub-arguments), giving a more refined persistence-based defeat criteria. Thus, an argument $\mathcal{A}$ using persistence is not necessarily defeated if attacked by a persistence-free argument $\mathcal{B}$; for example, if the persistent literals $\langle \ell, t \rangle, \ldots, \langle \ell, t+k \rangle$ in $\mathcal{A}$ are not part of the explanation of the other argument (e.g. if none these literals is in the sub-argument of $\mathcal{A}$ directly attacked by $\mathcal{B}$).

In contrast to the TDR system in (as well as other temporal argumentation systems [37] [17]), in t-DeLP we let the notion of an *interval where some conclusion holds* to be a notion deriving from the *set of time-points for which this conclusion holds*. In this sense, TDR is more expressive than t-DeLP, though for most applications, it seems possible to translate a TDR-proof for the warrant of $\langle \ell, [t, t'] \rangle$ in a given TDR program, into a t-DeLP-proof for the warrant of each $\langle \ell, t_0 \rangle$ with $t \leq t_0 \leq t'$ in a corresponding t-DeLP program. Figure 9 summarizes the differences between DeLP, t-DeLP and TDR.

Conclusions and Future Work

In this paper we have defined t-DeLP, a temporal version of DeLP where logic programs contain temporal literals and rules with duration. The proposed framework modifies features of DeLP in order to deal with specific issues related to temporal reasoning, like persistence and the past/future asymmetry in causal statements.

Features	DeLP	t-DeLP	TDR
literals	p or $\sim p$	$\langle p, t\rangle$ or $\langle \sim p, t\rangle$	$(\neg)\mathrm{Holds}_{at}(p, t)$ $(\neg)\mathrm{Holds}_{in}(p, [t, t'])$
rules	literal $\prec$ set of literals	literal $\prec$ set of literals	literal $\prec$ set of literals
derivability	modus ponens	modus ponens	modus ponens
argument $\mathcal{A}$	$\mathcal{A} \subseteq \Delta$ $\mathcal{A} \cup \Pi \vdash \ell$	$\mathcal{A} \subseteq \Pi \cup \Delta$ $\mathcal{A} \vdash \ell$	$\mathcal{A} \subseteq \Delta$ $\mathcal{A} \cup \Pi \vdash \ell$
$\mathcal{A}$ attacks $\mathcal{B}$	$\langle \mathcal{A}, \ell\rangle$ attacks $\langle \mathcal{B}, \ell'\rangle$ iff $\{\ell, \ell'\} \cup \Pi$ is inconsistent	$\sim\mathsf{concl}(\mathcal{A}) \in$ $\mathsf{literals}[\mathcal{B}]$	$\mathrm{Holds}_{in}(p, I)$ vs. $\neg\mathrm{Holds}_{in}(p, I')$, with $I \cap I' \neq \emptyset$
specificity	generalized spec. (activation sets)	Definition 7 (base = act. set, less persistence)	pointwise gen. spec. (pointwise activation sets)
more direct rules	by specificity	no	by specificity
more premises	if more specific	always	if more specific
less use of persistence	n/a	local comparison	global comparison
computing warrant	dialectical tree	dialectical tree	algorithm in [5]

Fig. 9 A comparison of DeLP, t-DeLP and TDR.

Besides this differences at the definition of defeat, t-DeLP is essentially based on the same argumentation-based procedure that defines the notion of defeasible logical consequence, or warrant. This notion of the set of warranted literals of a program has been shown to satisfy the postulates of Direct Consistency and Sub-arguments. In addition, we have extended the basic framework to deal with programs with a class of mutex constraints, and we have shown that the modified notion of warrant allows us to extend the previous resuts to Direct Consistency, Indirect Consistency and Closure as well.

For future work, we would like to study a preference relation which takes into account the criterion for *temporally more precise* information. Another interesting extension might aim for evidence-based reasoning, from the presently observable effects to its presumable past causes. Other technical improvements, like generalizing the Indirect Consistency and Closure from mutex sets to arbitrary strict rules, seem also of interest.

Appendix: proofs for auxiliary results.

The proofs for the auxiliary results mentioned in the previous sections are presented here. We start giving an inductive definition for the notion of sub-argument (Def. 5). Given an argument $\mathcal{A}$ in some t-de.l.p. (Π, Δ) and a literal $\langle \ell, t\rangle \in \mathsf{literals}(\mathcal{A})$, the sub-argument of $\mathcal{A}$ for $\langle \ell, t\rangle$, denoted $\mathcal{A}(\langle \ell, t\rangle)$, is the set obtained by the following inductive construction:

if $\delta \in \mathcal{A}$ exists with $\mathsf{head}(\delta) = \langle \ell, t\rangle$,	then $\delta \in \mathcal{A}(\langle \ell, t\rangle)$
if $\delta \in \mathcal{A}(\langle \ell, t\rangle)$ and $\delta' \in \mathcal{A}$ exists with $\mathsf{head}(\delta') \in \mathsf{body}(\delta)$,	then $\delta' \in \mathcal{A}(\langle \ell, t\rangle)$

Proposition 1 Given some argument $\mathcal{A}$ and a literal $\langle \ell, t\rangle \in \mathsf{literals}(\mathcal{A})$, then $\mathcal{A}(\langle \ell, t\rangle)$ is unique.

Proof By induction on the complexity of $\mathcal{A}$.

(Base Case) Suppose that $\mathcal{A}$ is a strict fact $\mathcal{A} = \{\langle \ell, t\rangle\} \subseteq \Pi_{\mathsf{f}}$. Then, $\mathsf{literals}(\mathcal{A}) = \mathcal{A}$ so $\mathcal{A}(\langle \ell, t\rangle) = \mathcal{A}$ and it is the only sub-argument of $\mathcal{A}$ deriving $\langle \ell, t\rangle$. Hence it is unique.

(Ind. Case) Assume (Ind. Hyp.) that for any argument $\mathcal{A}$ with some $\delta \in \mathcal{A}$ such that $\mathsf{head}(\delta) = \mathsf{concl}(\mathcal{A})(= \langle \ell, t\rangle)$, we have $\mathcal{A}(\langle \ell', t'\rangle)$ is unique for each $\langle \ell', t'\rangle \in \mathsf{literals}(\mathcal{A})$. We check the unicity of the remaining case $\mathcal{A}(\mathsf{concl}(\mathcal{A})) = \mathcal{A}$. Suppose another sub-argument $\mathcal{B} \subseteq \mathcal{A}$ exists for $\langle \ell, t\rangle$. From $\mathcal{B} \subseteq \mathcal{A}$ and $\mathcal{B} \neq \mathcal{A}$, we infer the existence of some rule or literal $\delta' \in \mathcal{A} \smallsetminus \mathcal{B}$. (Case $\delta' \in \Pi$) If this δ' is a literal or a strict rule, then such $\delta' \in \mathcal{A}_\Pi$ shows that $\mathcal{A}$ does not satisfy, in a $\subseteq$-minimal way, that $\mathcal{A}_\Delta \cup \mathcal{A}_\Pi \vdash \langle \ell, t\rangle$; hence $\mathcal{A}$ violates condition (4) from Def. 4, so $\mathcal{A}$ is not an argument (contradiction). (Case $\delta' \in \Delta$) Then δ' shows that $\mathcal{A}_\Delta$ does not satisfy $\Pi \cup \mathcal{A}_\Delta \vdash \langle \ell, t\rangle$ in a $\subseteq$-minimal way; so $\mathcal{A}$ does not satisfy condition (3) from Def.4. Again, $\mathcal{A}$ cannot be an argument (contradiction). □

Proposition 2 The following hold for any t-DeLP program:

(1) If $\mathcal{A}_1$ is a proper defeater for an argument $\mathcal{A}_0$ at $\mathcal{B}$, then $\mathcal{B}$ is not a defeater for $\mathcal{A}_1$.
(2) If $\mathcal{A}, \mathcal{B}$ attack each other, and $\mathcal{B}$ is not a proper defeater for $\mathcal{A}$, then $\mathcal{A}$ is a defeater for $\mathcal{B}$.

Proof For (1). Let $\mathcal{A}_1$ attack $\mathcal{A}_0$ at $\mathcal{B} = \mathcal{A}_0(\sim\mathsf{concl}(\mathcal{A}_1))$. Say $\mathsf{concl}(\mathcal{A}_1)) = \langle \ell, t\rangle$, so we must have $\langle \sim\ell, t\rangle = \mathsf{concl}(\mathcal{B})$. By definition, it cannot be the case that $\mathcal{B} \prec\succ \mathcal{A}_1$. So we must only check that $\mathcal{B} \not\succ \mathcal{A}_1$.
(Case $\mathsf{base}(\mathcal{A}_1) \supsetneq \mathsf{base}(\mathcal{B})$). First consider the condition: $\mathsf{base}(\mathcal{B}) \supsetneq \mathsf{base}(\mathcal{A}_1)$. This cannot be the case since otherwise, jointly with the case assumption $\mathsf{base}(\mathcal{A}_1) \supsetneq \mathsf{base}(\mathcal{B})$ we would infer that $\mathsf{base}(\mathcal{A}_1) \supsetneq \mathsf{base}(\mathcal{A}_1)$, which is impossible. The remaining condition to be ruled out is that $\mathcal{B}$ is longer than $\mathcal{A}_1$. But this cannot be the case either. To see this, assume the contrary: for some $t_0 < t$, $\mathcal{A}_1 = \mathcal{B}(\langle \ell, t_0\rangle) \cup \{\delta_\ell(t'')\}_{t_0 \leq t'' < t}$. Then, we have $\mathsf{base}(\mathcal{A}_1) = \mathsf{base}(\mathcal{B}(\langle \ell, t_0\rangle))$; this jointly with the fact that $\mathsf{base}(\mathcal{B}(\langle \ell, t_0\rangle)) \subseteq \mathsf{base}(\mathcal{B})$ gives $\mathsf{base}(\mathcal{A}_1) \subseteq \mathsf{base}(\mathcal{B})$, which contradicts the case assumption. Thus, in either case $\mathcal{B}$ cannot be a proper defeater for $\mathcal{A}_1$.

(Case $\mathcal{B} = \mathcal{A}_1(\langle \sim\ell, t'\rangle) \cup \{\delta_{\sim\ell}(t'')\}_{t' \leq t'' < t}$). Thus, $\mathsf{base}(\mathcal{B}) = \mathsf{base}(\mathcal{A}_1(\langle \ell, t'\rangle)$. The first case for $\mathcal{B} \succ \mathcal{A}_1$ is that $\mathsf{base}(\mathcal{B}) \supsetneq \mathsf{base}(\mathcal{A}_1)$. Since this and the latter fact would imply $\mathsf{base}(\mathcal{A}_1) \supseteq \mathsf{base}(\mathcal{A}_1(\langle \sim\ell, t'\rangle)) \supsetneq \mathsf{base}(\mathcal{A}_1)$, thus concluding that $\mathsf{base}(\mathcal{A}_1) \supsetneq \mathsf{base}(\mathcal{A}_1)$, which is impossible. The other possibility for $\mathcal{B} \succ \mathcal{A}_1$ to be ruled out is that $\mathcal{B}$ is longer than $\mathcal{A}_1$; this and the case assumption imply both $\mathcal{B}$ and $\mathcal{A}_1$ are longer than each other. Since $\mathcal{A}_1$ is longer than $\mathcal{B}$, some $\mathcal{A}_1 \smallsetminus \mathcal{A}_1(\langle \sim\ell, t'\rangle)$ contains a non-persistence rule δ. On the other hand, $\mathcal{B}$ being longer than $\mathcal{A}_1$ implies that the latter is of the form $\mathcal{B}(\langle \ell, t_0\rangle) \cup \{\delta_\ell(t''')\}_{t_0 \leq t''' < t}$. (Case $t_0 < t'$) This is incompatible with the previous existence of a non-persistence rule $\delta \in \mathcal{A}_1 \smallsetminus \mathcal{A}_1(\langle \sim\ell, t'\rangle)$. (Case $t_0 > t'$) Then $\mathsf{concl}(\mathcal{A}_1) = \langle \ell, t\rangle$ rather than $\langle \sim\ell, t\rangle$. (Case $t_0 = t'$) Then both $\langle \ell, t_0\rangle$ and $\langle \sim\ell, t_0$ are derivable from $\mathcal{A}_1$ contradicting that this is an argument.

Claim (2) is straightforward. Let $\mathcal{A}, \mathcal{B}$ be arguments attacking each other, and suppose $\mathcal{B} \not\succ \mathcal{A}$. (Case $\mathcal{A} \not\succ \mathcal{B}$) Then by definition, we have $\mathcal{A} \prec\succ \mathcal{B}$ so $\mathcal{A}$ is a defeater for $\mathcal{B}$. (Case $\mathcal{A} \succ \mathcal{B}$) Since a proper defeater is a defeater, we are also done. □

Lemma 1 For any t-de.l.p. (Π, Δ),

(1) If $[\mathcal{A}_1, \ldots, \mathcal{A}_k, \ldots, \mathcal{A}_n]$ is an argumentation line for $\mathcal{A}_1$, then $[\mathcal{A}_m, \ldots, \mathcal{A}_n]$ is an argumentation line for $\mathcal{A}_m$.
(2) Each argumentation line $\Lambda = [\mathcal{A}_1, \ldots] \in \mathcal{T}_{(\Pi,\Delta)}(\mathcal{A}_1)$ is finite. The dialectical tree $\mathcal{T}_{(\Pi,\Delta)}(\mathcal{A}_1)$ is finite.

Proof For (1), let $\Lambda_1 = [\mathcal{A}_1, \ldots, \mathcal{A}_m, \ldots, \mathcal{A}_n]$ be an argumentation line for $\mathcal{A}_1$. Notice that the first element of Λ_m is $\mathcal{A}_m$. We check that each condition (i)-(iii) from Definition 8 holds for the sequence $\Lambda_m = [\mathcal{A}_m, \ldots, \mathcal{A}_n]$.

(i) supporting (resp. interfering) arguments are jointly consistent.
The joint consistency of supporting (resp. interfering) arguments is satisfied by Λ_m, since otherwise if $\mathcal{A}_m \cup \ldots \cup \mathcal{A}_n \cup \Pi$ was inconsistent, then so would be $\mathcal{A}_1 \cup \ldots \cup \mathcal{A}_n \cup \Pi$, contradicting that Λ_1 is an arg. line.
(ii) If this condition failed for Λ_m at the pair $\mathcal{A}_{m+k} = \mathcal{A}_{m+k+2j}(\sim\mathsf{concl}(\mathcal{A}_{m+k+2j+1}))$, then it would already fail for Λ_1 at the same pair.
(iii) The condition that $\mathcal{A}_{i+1}$ is a proper defeater for $\mathcal{A}_i$ if $\mathcal{A}_i$ is a blocking defeater for $\mathcal{A}_{i-1}$ must hold for Λ_k since otherwise it would also fail for Λ_1 at the same triple $\mathcal{A}_{i-1}, \mathcal{A}_i, \mathcal{A}_{i+1}$.

For (2), let $\mathsf{concl}(\mathcal{A}_1) = \langle \ell, t \rangle$. Recall that $t < \omega$ and the set Lit is also finite, so the set of literals $\langle \ell', t' \rangle$ with $t' \leq t$ is finite. In consequence, the set of rules δ whose head is some $\langle \ell', t' \rangle$ is also finite. Since arguments are finite sets of rules and literals, the set of arguments $\mathcal{A}_{2n+1}$ whose conclusion is some $\langle \ell', t' \rangle$ with $t' \leq t$ is also finite. Hence each argumentation line for $\mathcal{A}_1$ is finite. Finally, there are a finite number of argumentation lines for $\mathcal{A}_1$ (again because the number of arguments for $t' \leq t$ are finite). The latter two facts imply that the dialectical tree $\mathcal{T}_{(\Pi,\Delta)}(\mathcal{A}_1)$ is finite. □

Corollary 1 Let $\Lambda = [\mathcal{A}_1, \mathcal{A}_2, \ldots, \mathcal{A}]$ be an argumentation line in $\mathcal{T}_{(\Pi,\Delta)}(\mathcal{A}_1)$. Then

(1) if $\mathcal{A} = \mathcal{A}_{2n+1}$ is undefeated in Λ, then in the corresponding arg. line $[\mathcal{A}_2, \ldots, \mathcal{A}]$ the (now interfering) argument $\mathcal{A}$ is undefeated;
(2) if $\mathcal{A} = \mathcal{A}_{2n}$ is defeated in Λ, then in the corresponding arg. line $[\mathcal{A}_2, \ldots, \mathcal{A}]$ the (now supporting) $\mathcal{A}$ defeated.

Proof (1) Given $\Lambda_{2n+1} = [\mathcal{A}_1, \mathcal{A}_2, \ldots, \mathcal{A}_{2n+1}]$, by Lemma 1 (1) we have $\Lambda'_{2n+1} = [\mathcal{A}_2, \ldots, \mathcal{A}_{2n+1}]$ is an arg. line. If $\mathcal{A}_{2n+1}$ is undefeated in Λ' we are done. Otherwise some $\mathcal{A}_{2n+2}$ exists with $\Lambda'_{2n+2} = [\mathcal{A}_2, \ldots, \mathcal{A}_{2n+1}, \mathcal{A}_{2n+2}]$ and $\mathcal{A}_{2n+2}$ evaluated as undefeated in Λ'_{2n+2}. Then $\Lambda_{2n+2} = \Lambda_{2n+1} {}^{\frown}[\mathcal{A}_{2n+2}]$ is an arg. line, and $\mathcal{A}_{2n+2}$ must be defeated there, since $\mathcal{A}_{2n+1}$ is undefeated. So some $\mathcal{A}_{2n+3}$ exists such that $\Lambda_{2n+3} = \Lambda_{2n+2} {}^{\frown}[\mathcal{A}_{2n+3}]$ is an arg. line and $\mathcal{A}_{2n+3}$ is undefeated there. This procedure can be repeated, as before generating an infinite sequence of increasing argumentation lines, which is impossible.

(2) The proof is analogous: let $\Lambda_{2n} = [\mathcal{A}_1, \ldots, \mathcal{A}_{2n}]$ be an arg. line with $\mathcal{A}_{2n}$ defeated. Then some $\mathcal{A}_{2n+1}$ exists with $\Lambda_{2n+1} = \Lambda_{2n} {}^{\frown}[\mathcal{A}_{2n+1}]$ and $\mathcal{A}_{2n+1}$ undefeated there. On the other hand, clearly $\Lambda'_{2n} = [\mathcal{A}_2, \ldots, \mathcal{A}_{2n}]$ is an arg. line, so if $\mathcal{A}_{2n}$ is defeated there we are done. Otherwise, some $\mathcal{A}_{2n+1}$ exists with $\Lambda'_{2n+1} = \Lambda'_{2n} {}^{\frown}[\mathcal{A}_{2n+1}]$ and $\mathcal{A}_{2n+1}$ defeated there. Then some $\mathcal{A}_{2n+2}$ exists with $\Lambda'_{2n+2} = \Lambda'_{2n+1} {}^{\frown}[\mathcal{A}_{2n+2}]$ and $\mathcal{A}_{2n+2}$ undefeated there. Then, $\mathcal{A}_{2n+2} = \Lambda_{2n+1} {}^{\frown}[\mathcal{A}_{2n+2}]$ is an arg. line. Since we had $\mathcal{A}_{2n+1}$ is undefeated, $\mathcal{A}_{2n+2}$ is defeated. This procedure can be repeated, again giving an infinite sequence of increasing arg. lines, which was shown to be impossible. □

Acknowledgements The authors would like to thank the anonymous reviewers for their helpful comments. This work has been partially supported by the Spanish MICINN projects CONSOLIDER-INGENIO 2010 Agreement Technologies CSD2007-00022 and ARINF TIN2009-14704-C03-03, with FEDER funds of the EU, and by the Generalitat de Catalunya grant 2009-SGR-1434.

References

1. M. Abadi and Z. Manna. Temporal logic programming. *Proc. of International Symposium on Logic Programming*, pp. 4–16, San Francisco, USA (1987)
2. G. Antoniou, M. Maher and D. Billington. Defeasible logic versus logic programming without negation as failure, *Journal of Logic Programming*, 42(1):47–57 (2000)
3. L. Amgoud and C. Cayrol. A reasoning model based on the production of acceptable arguments, *Annals of Mathematics and Artificial Intelligence* 34:197–215 (2002).
4. L. Amgoud and C. Cayrol. Inferring from inconsistency in preference-based argumentation frameworks, *International Journal of Automated Reasoning*, 29(2):125–169 (2002)
5. J. Augusto and G. Simari. Temporal Defeasible Reasoning, *Knowledge and Information Systems*, 3:287–318 (2001)
6. P. Besnard and A. Hunter. Argumentation Based on Classical Logic. Ch. 7 in [46]
7. D. Billington. Defeasible logic is stable. *Journal of Logic and Computation*, 3:379–400 (1993)
8. P. Blackburn, J. van Benthem and F. Wolter (eds.) *Handbook of Modal Logic*, Volume 3. Elsevier, New York, USA (2006)
9. G. Brewka. Adding priorities and specificity to default logic, *Proc. of the 4th European Workshop on Logic in Artificial Intelligence JELIA94*, LNAI 838, pp. 247260, Springer-Verlag, York, England (1994)
10. G. Brewka, I. Niemelä and M. Truszczyński. Nonmonotonic Reasoning, *Handbook of Knowledge Representation*, F. van Harmelen, V. Lifschitz and B. Porter (eds.), Ch 6. Elsevier, Oxford, UK (2007)
11. J. Broersen, R. Wieringa and J.-J. Meyer. A Semantics for Persistency in Propositional Dynamic Logic, In *Proceedings of 1st Conf. on Computation Logic (CL 2000)* J. Lloyd et al. (eds.), pp. 912–925 (2000)
12. M. Caminada and L. Amgoud. On the evaluation of argumentation formalisms, *Artificial Intelligence*, 171:286–310 (2007)
13. M. Capobianco, C. Chesñevar and G. Simari. Argumentation and the Dynamics of Warranted Beliefs in Changing Environments, *Journal of Autonomous Agents and Multi-Agent Systems*, 11:127–151 (2005)
14. M. Castilho, O. Gasquet and A. Herzig. Formalizing Action and Change in Modal Logic I: the Frame Problem, *Journal of Logic and Computation*, 9(5): 701–735 (1999)
15. B. Chellas. *Modal logic, an introduction*. Cambridge Univ. Press, Cambridge (1980)
16. C. Chesñevar, J. Dix, F. Stolzenburg and G. Simari. Relating defeasible and normal logic programming through transformation properties. *Theoretical Computer Science*, 290: 499–529 (2003)
17. L. Cobo, D. Martínez and G. Simari. Stable Extensions in Timed Argumentation Frameworks, *Theories and Applications of Formal Argumentation TAFA 2011*, pp. 181-196, Barcelona (2011)
18. L. Cobo, D. Martínez and G. Simari. Acceptability in Timed Frameworks with Intermittent Arguments, *Artificial Intelligence Applications and Innovations, AIAI 2011* Part II, pp. 202–211 (2011)
19. R. Craven and M. Sergot. Distant Causation in $\mathcal{C}+$, *Studia Logica*, 79:73–96, (2005)
20. T. Delladio and G. Simari. Relating DeLP and Default Logic, *Inteligencia Artificial*, 35: 101–109 (2007)

21. P. Dung. On the acceptability of arguments and its fundamental role in nonmonotonic reasoning, logic programming and n-person games* 1, *Artificial Intelligence*, 77(2):321–357 (1995)
22. E. Emerson. Temporal and modal logic, in J. van Leeuwen (ed.) *Handbook of Theoretical Computer Science*, pp. 996–1072, Elsevier, New York, USA (1990)
23. A. García and G. Simari. Defeasible logic programming: An argumentative approach, *Theory and Practice of Logic Programming*, 4(1+2): 95–138 (2004)
24. M. Gelfond and V. Lifschitz. Representing action and change by logic programs, *Journal of Logic Programming*, 17(2,3&4): 301–321(1993)
25. M. Ghallab, D. Nau and P. Traverso. *Automated Planning: Theory and Practice*. Morgan Kaufmann, San Francisco, USA (2004)
26. G. De Giacomo and M. Lenzerini. PDL-based framework for reasoning about actions, *Proceedings of the 4th Congress of the Italian Association for Artificial Intelligence IA*AI95*, pp. 103114. LNAI 992 (1995)
27. L. Giordano, A. Martelli and C. Schwind. Ramification and Causality in a Modal Action Logic, *Journal of Logic and Computation*, 10(5): 625–662 (2000)
28. E. Giunchiglia, J. Lee, V. Lifschitz, N. McCain and H. Turner. Non- monotonic causal theories, *Artificial Intelligence*, 153:49–104 (2004).
29. R. Goldblatt. *Logics of time and computation*. CSLI, Stanford, USA (1992)
30. G. Governatori and P. Terenziani. Temporal Extensions to Defeasible Logic, *Proc. of 20th Australian Joint Conference on Artificial Intelligence, AI 2007*, M. Orgun and J. Thornton (eds.), pp. 1–10, Springer (2007)
31. D. Harel, D. Kozen and J. Tiuryn *Dynamic Logic*. MIT Press, Massachusetts, USA (2000)
32. A. Hunter. Execution of defeasible temporal clauses for building preferred models. *Proc. of Fundamentals of Artificial Intelligence Research, FAIR '91* pp. 84–98 (1991)
33. A. Hunter. Merging structured text using temporal knowledge.*Data Knowledge Engineering*, 41(1): 29–66 (2002)
34. K. Konolige. On the relation between default and autoepistemic logic, *Artificial Intelligence*, 35:342–382 (1988)
35. R. Kowalski and M. Sergot, A logic-based calculus of events, *New Generation Computing*, 4:67–95 (1986)
36. D. Lewis *Counterfactuals*. Basil Blackwell, Oxford, UK (1973)
37. N. Mann and A. Hunter. Argumentation Using Temporal Knowledge. *Proc. of Computer Models of Argumentation (COMMA'08)*, pp. 204–215 IOS Press (2008)
38. J. McCarthy and P. Hayes. Some philosophical problems from the standpoint of artificial intelligence, *Machine Intelligence*, 4:463–502 (1969)
39. S. Modgil. Reasoning about preferences in argumentation frameworks, *Artificial Intelligence*, 173:901–934 (2009)
40. D. Nute. Defeasible Logic, in *Handbook of Logic in Artificial Intelligence and Logic Programming*, Vol. 3, pp. 353–395. Oxford Univ. Press, Oxford, UK (1994)
41. P. Pardo and L. Godo. t-DeLP: a temporal extension of the defeasible logic programming argumentative framework, *Proceedings of SUM 2011*, LNAI vol. 6929 Springer-Verlag, pp. 489-503, Dayton, USA (2011)
42. D. Poole. On the Comparison of Theories: Preferring the Most Specific Explanation, *Proceedings of 9th International Joint Conference in Artificial Intelligence*, pp. 144–147, Los Angeles, USA (1985)
43. H. Prakken. An abstract framework for argumentation with structured arguments. *Argument & Computation*, 1(2):93–124 (2010)
44. H. Prakken and G. Sartor. Argument-based extended logic programming with defeasible priorities. *Journal of Applied Non-classical Logics*, 7:25–75 (1997)
45. H. Prendinger and G. Schurz, Reasoning about action and change: a dynamic logic approach. *Journal of Logic, Language, and Information*, 5:209–245 (1996)
46. I. Rahwan and G. Simari (eds.) *Argumentation in Artificial Intelligence*. Springer (2011)
47. R. Reiter. A logic for default reasoning. *Artificial Intelligence* 13:81–132 (1980)
48. J. Rintanten. On Specificity in Default Logic. (Manuscript) 1995
49. F. Stolzenburg, A. García, C. Chesñevar and G. Simari. Computing Generalized Specificity. *Journal of Applied Non-Classical Logics*, 12(1):1–27 (2002)
50. M. Thimm and G. Kern-Isberner, On the Relationship of Defeasible Argumentation and Answer Set Programming. *Proc. of Computer Models of Argumentation (COMMA'08)* P. Besnard, S. Doutre, and A. Hunter (eds.) pp. 393–404. IOS Press, 2008.
51. D. Zhang and N. Foo. Frame problem in dynamic logic. *Journal of Applied Non-Classical Logics* 15(2)215–239 (2005)