# Distinguishing ground from nonground information in defeasible argumentation

Carlos I. Chesñevar[1] Guillermo R. Simari

Instituto de Ciencias e Ingeniería de Computación (ICIC)
Grupo de Investigación en Inteligencia Artificial (GIIA)
Departamento de Ciencias de la Computación
Universidad Nacional del Sur
Av.Alem 1253 – (8000) Bahía Blanca – REPÚBLICA ARGENTINA
FAX: (54) (91) 553933 – PHONE: (54) (91) 20776 (ext.208) – Email: **ccchesne@criba.edu.ar**

KEYWORDS: artificial intelligence, defeasible reasoning, argumentative systems

**Abstract**

The problem of speeding up inference has proved to be important in argumentative systems. When computing dialectical structures, several paths are searched (called *argumentation lines*), many of which will eventually prove useless. Morover, the performance of backward chainers degrades quickly as the size of the knowledge base increases. This is a major hindrance for argumentative systems, since backward chaining is applied at two different levels (arguments themselves resemble proof trees, and arguments are related to each other within a tree structure).

This paper presents a novel approach for speeding up inference. We will work with *arguments for nonground literals*, on the basis of some results presented by Levy [2]. As a result, most computations involving dialectical trees can be performed independently from the current ground facts in the system's knowledge base. The definition of *abstract dialectical tree* will be introduced, which will account for *all* nonground queries for a given literal.

[1]Supported by a fellowship of the Consejo Nacional de Investigaciones Científicas y Técnicas (CONICET), República Argentina.

# Distinguishing ground from nonground information in defeasible argumentation

## 1 Introduction and motivations

In *A Mathematical Treatment of Defeasible Reasoning* [6], or *MTDR*, a clear and theoretically sound structure for an argument-based reasoning system was introduced. Ever since, many aspects of that framework have been the focus of research, producing some interesting results [5, 4]. Within *MTDR*, an argument $A$ for a hypothesis $h$ represents a tentative piece of reasoning an intelligent agent would be inclined to accept, all things considered, as an *explanation* for $h$. The argument $A$ may be defeated in presence of counterarguments, which on their turn may be defeated by counter-counterarguments, and so on. As a consequence, several arguments may be *raised*, *defeated*, and *reinstated* in order to determine whether the original argument $A$ is accepted. This defines a tree structure, called *dialectical tree.*

Speeding up inference has proved to be an important issue in argumentative systems, since the construction of dialectical trees is computationally expensive: consistency checking within arguments and comparisons using a preference criterion (specificity) are involved. When computing dialectical trees, the argumentative inference engine considers several paths (called *argumentation lines*), many of which will eventually prove useless. Morover, the performance of a backward chainer degrades quickly as the size of the knowledge base increases. This is a major hindrance for argumentative systems, since backward chaining is applied at two different levels: on the one hand, arguments themselves resemble proof trees; on the other hand, arguments are related to each other within a tree structure. In order to solve all these problems, the notion of 'arguments base' was first developed [1], from which some correspondences between *Truth Maintenance Systems* and Defeasible Argumentation could be observed.

In this paper, we will take a different approach for speeding up inference. As a basis, we adopt some ideas presented by Levy in [2]. He introduces the concept of *query tree*, which encodes *all* possible derivations for a given query. That way, incorporating new ground facts do not affect the query tree, so that its computation may be amortized by many queries. We proceed in a similar way, even though our approach to computing arguments differs greately from Levy's, since inference in our framework is defeasible, and hence non-monotonic. Instead of working with arguments for ground literals, (built *dynamically* as the corresponding dialectical tree is generated), we will compute *argument schemata for nonground literals.* In fact, we will be able to compute a dialectical tree structure relating several argument schemata, which can be later instanciated to particular ground facts. As a result, most computations involving dialectical trees can be performed independently from the current ground facts in the system's knowledge base. In order to do so, we will extend the original *MTDR* framework, so that we can deal with relations among nonground argumentation sequences. This will enable us to compute dialectical tree structures, called *abstract dialectical trees*, accounting for *all* queries associated with a nonground literal.

## 2 Abstract derivations and argument schemata

We refer those readers not familiar with the *MTDR* framework to the appendix, at the end of this paper. A complete definition of the framework can be also found elsewhere [6]. The notions of *argument*, *counterargument* and *defeat* can be seen in the following example, [2] which will be used as motivation for introducing the definition of *argument schemata.*

EXAMPLE **2.1** (Vehicles are typically not allowed in parks)[3] The following knowledge base contains some information about what a vehicle is, and what kind of vehicles should be allowed in a park. (Note: *allowed* stands for *allowed_in_the_park*)

$$\begin{aligned}\Delta \quad = \quad & \{\ vehicle(X) \succ\!\!- \neg\ allowed(X),\\ & vehicle(X) \wedge used\_for\_emergencies(X) \succ\!\!- allowed(X),\\ & has\_wheels(X) \wedge has\_motor(X) \succ\!\!- vehicle(X),\\ & has\_siren(X) \succ\!\!- used\_for\_emergencies(X),\\ & has\_siren(X) \wedge off\_duty(X) \succ\!\!- \neg\ used\_for\_emergencies(X),\\ & vehicle(X) \wedge for\_park\_maintenance(X) \succ\!\!- allowed(X)\\ & toy(X) \succ\!\!- \neg\ vehicle(X)\ \}\end{aligned}$$

$$\mathcal{K}_G \quad = \quad \{human(X) \rightarrow \neg\ vehicle(X)\}$$

Let $\mathcal{K}_P$=
{ *human*(**gardener**), *has_wheels*(**wheelbarrow**), *has_wheels*(**sports-car**), *has_motor*(**sports-car**), *has_wheels*(**ambulance**), *has_motor*(**ambulance**), *has_siren*(**ambulance**), *has_motor*(**toy-car**), *has_wheels*(**toy-car**), *toy*(**toy-car**) }.

Then $A$= { *has_wheels*(**ambulance**) $\wedge$ *has_motor*(**ambulance**) $\succ\!\!-$ *vehicle*(**ambulance**), *vehicle*(**ambulance**) $\succ\!\!-$ $\neg$*allowed*(**ambulance**)}
is an argument for $\neg$*allowed*(**ambulance**),

and $B$ =
{*has_wheels*(**ambulance**) $\wedge$ *has_motor*(**ambulance**) $\succ\!\!-$ *vehicle*(**ambulance**),
*has_siren*(**ambulance**) $\succ\!\!-$ *used_for_emergencies*(**ambulance**),
*vehicle*(**ambulance**) $\wedge$ *used_for_emergencies*(**ambulance**) $\succ\!\!-$ *allowed*(**ambulance**) }
is an argument for *allowed*(**ambulance**).

Then $\langle B, allowed(\textbf{ambulance})\rangle$ defeats $\langle A, \neg allowed(\textbf{ambulance})\rangle$, since the first argument counterargues (and is more specific than) the second. There is no argument structure for $\neg$*allowed*(**wheelbarrow**) (since there is no enough information for concluding anything about allowance of **wheelbarrow**).

Next we will introduce some new definitions, which will allow us to extend *MTDR* for dealing with nonground argument-like structures. These structures will be called

[2]For the sake of clarity, predicate letters will be denoted using lowercase italics, *e.g. flies*; constants will be denoted using lowercase sans-serif letter style, *e.g.*tweety, opus.

[3]This example was originally presented by R.Loui in [3].

*argument schemata*, since they may be later instanciated to a number of different *argument structures*. Firstly, we will extend the pair $(\mathcal{K}, \Delta)$ to a 3-uple $(\mathcal{K}, \Delta, \mathcal{P})$, distinguishing a set $\mathcal{P}$ of *base predicates*, which characterize basic input information. Predicates appearing in $\mathcal{K}_P$ will be a subset of those included in $\mathcal{P}$.

EXAMPLE **2.2** Consider example 2.1 above. We will take $\mathcal{P}$ to be the set {*toy*, *human*, *has_wheels*, *has_motor*, *has_siren*, *for_park_maintenance*, *off_duty*}.

Let $\overline{X}$ denote a vector of variables $(X_1, X_2, \ldots, X_k)$. Analogously, $\overline{C}$ denotes a vector of constants $(c_1, c_2, \ldots, c_k)$. For ease of reading, parentheses will be dropped when referring to a single constant. Let $q(\overline{W})$ [4] be a nonground literal of arity $n$ (we will also write $arity(q) = n$). Then $q(\overline{W})^{\downarrow}$ (or simply $q^{\downarrow}$) denotes a ground instance of $q$. If $\overline{C} = (c_1, c_2, \ldots, c_n)$ is a vector of constants, $q^{\downarrow\overline{C}}$ denotes the ground instance $q(\overline{C}) = q(c_1, c_2, \ldots, c_n)$.

## 2.1 Abstract derivations

Abstract derivations[5] allow us to perform inference from nonground queries, using backward chaining. Through unification we propagates variable names from the nonground query to the leaves of the proof tree. Next we will state the formal definition of *abstract derivation*:

DEFINITION **2.1** *Abstract (defeasible) derivation* Given a knowledge base $(\mathcal{K}, \Delta, \mathcal{P})$, let $Rules \subseteq \mathcal{K}_G$, let $DefeasibleRules \subseteq \Delta$, let $Facts \subseteq \mathcal{P}$ and let $q(\overline{X})$ be a nonground literal. An *abstract defeasible derivation for* $q(\overline{X})$ from $Rules \cup DefeasibleRules \cup Facts$, denoted $Rules \cup DefeasibleRules \cup Facts \stackrel{a}{\sim} q(\overline{X})$ is defined recursively as follows:

- There exists some predicate $p \in Facts$, such that $p = q$ and $arity(p) = arity(q)$. The nonground literal $p(\overline{X})$ will be called *activation literal.*
- There exists a rule $a_1 \wedge a_2 \wedge \ldots a_k \rightarrow b \in Rules$ and an renaming sustitution $\theta$ such that $b\theta = q(\overline{X})$ and there exists an abstract derivation for $a_1\theta$, $a_2\theta$, ..., $a_k\theta$.
- There exists a rule $a_1 \wedge a_2 \wedge \ldots a_k \succ\!\!- b \in DefeasibleRules$ and an renaming sustitution $\theta$ such that $b\theta = q(\overline{X})$, and there exists an abstract derivation for $a_1\theta$, $a_2\theta$, ..., $a_k\theta$.

The set $\{p_1(\overline{X_1}), p_2(\overline{X_2}), \ldots, p_n(\overline{X_n})\}$ of all activation literals will be called *activation set.* If there are no applications of the third case in the abstract derivation of $q$, this can be denoted as $Rules \cup Facts \stackrel{a}{\vdash} q$. □

[4] When no ambiguity arises, we will simply write $q$.

[5] Levy uses the term "symbolic derivations" (see [2]), because it is normally used in deductive databases (A.Levy, personal communication). However, we consider the word "symbolic" too generic for our context.

As we can see in this definition, we consider rule chaining as usual in backward reasoning. We just impose the restriction of using a nonground query as the first goal to be solved. We can chain nonground defeasible rules in order to *explain* some nonground literal. Chained rules, satisfying certain constraints, constitute an *argument schema.*

DEFINITION **2.2** Let $A \subseteq \Delta$ be a set of defeasible rules, and let $h(\overline{X})$ be a nonground literal. Then $A$ will be an *argument schema* for $h(\overline{X})$, denoted $\langle\langle A, h(\overline{X})\rangle\rangle$, iff

a) There exists an activation set $S \subseteq \mathcal{P}$ such that $\mathcal{K}_G \cup S \cup A \mid\!\sim^{a} h(\overline{X})$ (explanation).

b) Let $R \subseteq \mathcal{K}_G \cup A$ be the set of rules used in the derivation of $h$. Then there are no complementary literals $p(\overline{X})$ and $\neg p(\overline{X})$ in $R$ (internal coherence).

c) There exists no $A' \subset A$, such that $\mathcal{K}_G \cup A' \cup S \mid\!\sim^{a} h(\overline{X})$. (minimality).

A *subargument schema* of $\langle\langle A, h(\overline{X})\rangle\rangle$ is an argument schema $\langle\langle B, q(\overline{W})\rangle\rangle$ such that $B \subseteq A$. □

EXAMPLE **2.3** Consider example 2.1. Then
$A$=$\{has\_wheels(X) \wedge has\_motor(X) \succ\!\!- vehicle(X),\ vehicle(X) \succ\!\!- \neg allowed(X)\}$ is an argument schema for $\neg allowed(X)$.

It should be remarked that definitions 2.3 and 2.4 are similar to defs. A.3 and A.4 for argument structures. However, we must note that counterargumentation has been defined in terms of complementary literals instead of inconsistency.

DEFINITION **2.3** Given two argument schemata $\langle\langle A_1, h_1(\overline{X_1})\rangle\rangle$ and $\langle\langle A_2, h_2(\overline{X_2})\rangle\rangle$, we say that $\langle\langle A_1, h_1(\overline{X_1})\rangle\rangle$ *counterargues* $\langle\langle A_2, h_2(\overline{X_2})\rangle\rangle$ iff there exists a subargument schema $\langle\langle S, h(\overline{Z})\rangle\rangle$, where $S \subseteq A_2$, and such that $h_1(\overline{X_1})$ and $h(\overline{Z})$ are complementary literals. □

DEFINITION **2.4** Let $\mathcal{D} = \{a \in \mathcal{L} : a(\overline{X})$ is a nonground literal and $\mathcal{K}_G \cup \Delta \cup \mathcal{P} \mid\!\sim^{a} a(\overline{X})\}$, and let $\langle\langle A_1, h_1(\overline{X_1})\rangle\rangle$ and $\langle\langle A_2, h_2(\overline{X_2})\rangle\rangle$ be two argument schemata. We say that $\langle\langle A_1, h_1(\overline{X_1})\rangle\rangle$ is *strictly more informed than* $\langle\langle A_2, h_2(\overline{X_2})\rangle\rangle$, denoted $\langle\langle A_1, h_1(\overline{X_1})\rangle\rangle \succ_{\mathbf{info}} \langle\langle A_2, h_2(\overline{X_2})\rangle\rangle$, if and only if
i) $\forall S \subseteq \mathcal{D}$ if $\mathcal{K}_G \cup S \cup A_1 \mid\!\sim^{a} h_1(\overline{X_1})$ and $\mathcal{K}_G \cup S \not\mid\!\sim h_1(\overline{X_1})$, then $\mathcal{K}_G \cup S \cup A_2 \mid\!\sim^{a} h_2(\overline{X_2})$.
ii) $\exists S \subseteq \mathcal{D}$ such that $\mathcal{K}_G \cup S \cup A_2 \mid\!\sim^{a} h_2(\overline{X_2})$, $\mathcal{K}_G \cup S \not\mid\!\sim h_2(\overline{X_2})$ and $\mathcal{K}_G \cup S \cup A_1 \not\mid\!\sim h_1(\overline{X_1})$.
□

DEFINITION **2.5** Given two argument schemata $\langle\langle A_1, h_1(\overline{X_1})\rangle\rangle$ and $\langle\langle A_2, h_2(\overline{X_2})\rangle\rangle$, we say that $\langle\langle A_1, h_1\rangle\rangle$ *defeats* $\langle\langle A_2, h_2\rangle\rangle$ iff $\langle\langle A_1, h_1(\overline{X_1})\rangle\rangle$ counterargues $\langle\langle A_2, h_2\rangle\rangle$, and the counterargumentation subargument $\langle\langle S, \neg h_1(\overline{X_1})\rangle\rangle$ is such that
1. $\langle\langle A_1, h_1(\overline{X_1})\rangle\rangle$ is *more informed than* $\langle\langle S, \neg h_1(\overline{X_1})\rangle\rangle$, or 2. $\langle\langle A_1, h_1(\overline{X_1})\rangle\rangle$ is unrelated by $\succ_{\mathbf{info}}$ to $\langle\langle S, \neg h_1(\overline{X_1})\rangle\rangle$.
□

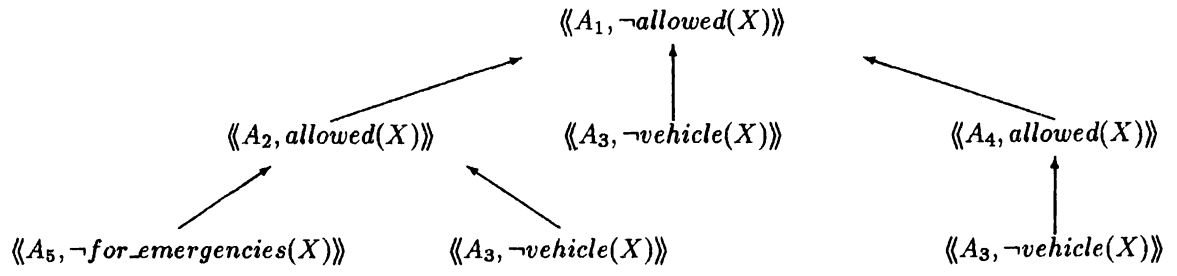


Figure 1: An abstract dialectical tree for $\langle\langle A_1, \neg allowed(X)\rangle\rangle$

In order to determine whether an argument schema supports an argument for a grounded literal $h(\overline{C})$, constants in $\overline{C}$ are propagated backward in the argument schema through unification, until a base predicate $p_i$ and some vector of constants $C_i$ is reached. If $p_i(C_i)$ belongs to $\mathcal{K}_P$, for every base predicate $p_i$ in $A$, and the consistency condition holds (see condition 2 in def. A.2), then we say that $\langle\langle A, h(\overline{X})\rangle\rangle$ *supports an argument for* $h(\overline{C})$. Thus we get an (instanciated) argument structure from a (noninstanciated) argument schema.

DEFINITION **2.6** Let $\langle\langle A, h(\overline{X})\rangle\rangle$ be an argument schema, and let $\overline{C} = (c_1, c_2, \ldots, c_k)$ be a vector of constants in $\mathcal{L}$. We will say that $\langle\langle A, h(\overline{X})\rangle\rangle$ *supports an argument for* $h(\overline{C})$ iff there exists a set $A' \subseteq A^{\downarrow}$ such that $\langle A', h(\overline{C})\rangle$. We will also write $\langle\langle A, h(\overline{X})\rangle\rangle$ *supports* $\langle A', h(\overline{C})\rangle$.[6] □

EXAMPLE **2.4** Consider the knowledge base in example 2.1. Then $\langle\langle A_1, \neg allowed(\overline{X})\rangle\rangle$ supports an argument for $\neg allowed$(**sports-car**), but $\langle\langle A_1, \neg allowed(\overline{X})\rangle\rangle$ does *not* support an argument for $\neg allowed$(**wheelbarrow**) (since no argument for $\neg allowed$(**wheelbarrow**) using the rules in $A_1$ can be built).

There is an interesting correspondence between the relations $\succ_{\textbf{info}}$ and $\succ_{\textbf{spec}}$, expressed through the following preposition

PROPOSITION **2.1** Let $\langle\langle A_1, h_1\rangle\rangle$, $\langle\langle A_2, h_2\rangle\rangle$ be two argument schemata, such that $\langle\langle A_1, h_1\rangle\rangle$ $\succ_{\textbf{info}}$ $\langle\langle A_2, h_2\rangle\rangle$. Let $\overline{C_1}$ and $\overline{C_2}$ be two vectors of constants, such that $\langle A_1', h_1(\overline{C_1})\rangle$ and $\langle A_2', h_2(\overline{C_2})\rangle$) are argument structures supported by by $\langle\langle A_1, h_1\rangle\rangle$ and $\langle\langle A_2, h_2\rangle\rangle$, respectively. Then either $\langle A_1', h_1(\overline{C_1})\rangle$ $\succ_{\textbf{spec}}$ $\langle A_2', h_2(\overline{C_2})\rangle$, or both argument structures are unrelated by specificity. □

From this proposition, we see that the $\succ_{\textbf{info}}$ relation subsumes the two alternative conditions (see def. A.5) necessary for considering an argument structure as a defeater. Thus, an abstract dialectical tree can be built, in which each node (argument schema) is related to its parent via the $\succ_{\textbf{info}}$ relation. After instanciating argument schemata to particular argument structures, the structure of the tree will be preserved, and a dialectical tree (see def. A.6) will result.

[6]Note that defeasible rules in $\langle\langle A, h\rangle\rangle$ are nonground, whereas argument structures such as $\langle A', h(\overline{C})\rangle$ involve ground defeasible rules.

DEFINITION **2.7** From the definitions discussed above, an *abstract dialectical tree* $\mathcal{T}_{\langle A,h\rangle}$ for an argument schema $\langle\langle A,h\rangle\rangle$ comes out to be recursively defined as follows:

1. A single node containing an argument schema $\langle\langle A,h\rangle\rangle$ with no defeaters is by itself an abstract dialectical tree for $\langle\langle A,h\rangle\rangle$. This node is also the root of the tree.
2. Let $\langle\langle A,h\rangle\rangle$ be an argument schema with defeaters $\langle\langle A_1,h_1\rangle\rangle$, $\langle\langle A_2,h_2\rangle\rangle$, ..., $\langle\langle A_n,h_n\rangle\rangle$. Then the abstract dialectical tree for $\mathcal{T}_{\langle A,h\rangle}$ can be obtained by putting $\langle\langle A,h\rangle\rangle$ as the root node of $\mathcal{T}_{\langle A,h\rangle}$ and by making this node the parent node of the roots of the abstract dialectical trees $\mathcal{T}_{\langle A_1,h_1\rangle}$, $\mathcal{T}_{\langle A_2,h_2\rangle}$, ..., $\mathcal{T}_{\langle A_n,h_n\rangle}$.

□

EXAMPLE **2.5** Consider the argument schemata $\langle\langle A_1, \neg allowed(X)\rangle\rangle$, $\langle\langle A_2, allowed(X)\rangle\rangle$, $\langle\langle A_3, \neg vehicle(X)\rangle\rangle$, $\langle\langle A_4, allowed(X)\rangle\rangle$ and $\langle\langle A_5, \neg for\_emergencies(X)\rangle\rangle$ in example 2.1, where

$A_1$ = $\{has\_wheels(X) \wedge has\_motor(X) \succ\!\!- vehicle(X),\ vehicle(X) \succ\!\!- \neg allowed(X)\ \}$.
$A_2$ = $\{has\_wheels(X) \wedge has\_motor(X) \succ\!\!- vehicle(X),\ has\_siren(X) \succ\!\!- for\_emergencies(X),$
$vehicle(X) \wedge for\_emergencies(X) \succ\!\!- allowed(X)\ \}$
$A_3$ = $\{toy(X) \succ\!\!- \neg vehicle(X)\ \}$
$A_4$ = $\{has\_wheels(X) \wedge has\_motor(X) \succ\!\!- vehicle(X),$
$vehicle(X) \wedge for\_park\_maintenance(X) \succ\!\!- allowed(X)\ \}$
$A_5$ = $\{has\_siren(X) \wedge off\_duty(X) \succ\!\!- \neg for\_emergencies(X)\}$

The abstract dialectical tree for $\langle\langle A_1, \neg allowed(X)\rangle\rangle$ is shown in figure 1

Next we list an algorithm, which allows us to get a dialectical tree out of an abstract dialectical tree.

**Algorithm** GetDialecticalTree
**Input:** abstract dialectical tree $\mathcal{T}_{\langle A,h(\overline{X})\rangle}$ and a vector of constants $\overline{C}$.
**Output:** dialectical tree $\mathcal{T}_{\langle A',h(\overline{C})\rangle}$ (if it exists), where $\langle A',h(\overline{C})\rangle$ is supported by $\langle\langle A,h(\overline{X})\rangle\rangle$
**If** $\mathcal{T}_{\langle A,h(\overline{X})\rangle}$ has a single node
**then**
**If** $\langle\langle A,h(\overline{X})\rangle\rangle$ supports an argument $A'$ for $h(\overline{C})$
**then**
Root of $\mathcal{T}_{\langle A,h(\overline{C})\rangle} = \langle A',h(\overline{C})\rangle$
**else**
Let $\mathcal{T}_{\langle A_1,h(\overline{X_1})\rangle}$, $\mathcal{T}_{\langle A_2,h(\overline{X_2})\rangle}$, ..., $\mathcal{T}_{\langle A_k,h(\overline{X_k})\rangle}$ be the immediate subtrees of $\mathcal{T}_{\langle A,h(\overline{X})\rangle}$.
**For every** $\langle\langle A_i,h(\overline{X_i})\rangle\rangle$
Let $p_i(\overline{C_i})$ be the counterargumentation literal in $\langle A',h(\overline{C})\rangle$.
GetDialecticalTree for every $\langle\langle A_i,h(\overline{X_i})\rangle\rangle$ using vector of constants $\overline{C_i}$.

## 2.2 Activating argument schemata

Whenever an argument schema $\langle\langle A, h(\overline{X})\rangle\rangle$ supports an argument structure $\langle A', h(\overline{C})\rangle$, a necessary condition is that every every literal in the activation set of $\langle\langle A, h(\overline{X})\rangle\rangle$ belongs to $\mathcal{K}_P$.[7]

DEFINITION **2.8** Let $\langle\langle A, h\rangle\rangle$ be an argument schema, and let $S = \{p_1, p_2, \ldots, p_k\}$ be its associated activation set. Then $f(\langle\langle A, h\rangle\rangle) = p_1 \wedge p_2 \wedge \ldots \wedge p_k$ will be the *activation formula* for $\langle\langle A, h\rangle\rangle$.

If $\langle\langle A, h\rangle\rangle$ supports $\langle A', h(\overline{C})\rangle$, and $\langle A', h(\overline{C})\rangle$ has $S = p_1(\overline{C_1}), p_2(\overline{C_2}), \ldots p_k(\overline{C_k})$ as activation set), then $f(\langle A', h(\overline{C})\rangle) = p_1(\overline{C_1}) \wedge p_2(\overline{C_2}) \wedge \ldots \wedge p_k(\overline{C_k})$ denotes the *activation formula* for $\langle A, h(\overline{C})\rangle$ □

We can also extend the definition of activation formula for an argument schema, in order to deal with abstract dialectial trees. Every argument schema in an abstract dialectical tree will be actived whenever its activation set belongs to $\mathcal{K}_P$, and neither of its associated defeaters is active.

DEFINITION **2.9** Let $\mathcal{T}_{\langle A, h\rangle}$ be an abstract dialectical tree for $\langle\langle A, h\rangle\rangle$. The *activation formula* for $\mathcal{T}_{\langle A, h\rangle}$, denoted $f(\mathcal{T}_{\langle A, h\rangle})$, can be defined recursively as follows:

a) If $\mathcal{T}_{\langle A, h\rangle}$ has a single node, then $f(\mathcal{T}_{\langle A, h\rangle}) = f(\langle\langle A, h\rangle\rangle)$.

b) Let $\mathcal{T}_{\langle A_1, h_1\rangle}$, $\mathcal{T}_{\langle A_2, h_2\rangle}$, $\ldots$, $\mathcal{T}_{\langle A_n, h_n\rangle}$ be the immediate subtrees of $\mathcal{T}_{\langle A, h\rangle}$.
Then $f(\mathcal{T}_{\langle A, h\rangle}) = f(\langle\langle A, h\rangle\rangle) \wedge \neg$ ( $f(\mathcal{T}_{\langle A_1, h_1\rangle}) \vee f(\mathcal{T}_{\langle A_2, h_2\rangle}) \vee \ldots \vee f(\mathcal{T}_{\langle A_n, h_n\rangle})$ )

If $\langle A', h(\overline{C})\rangle$ is an argument structure supported by $\langle\langle A, h\rangle\rangle$, then the activation formula for $f(\mathcal{T}_{\langle A', h(\overline{C})\rangle})$ is defined analogously. □

EXAMPLE **2.6** Consider example 2.5. Then the activation formula for $\mathcal{T}_{\langle A_1, \neg allowed(X)\rangle}$ is $(has\_wheels(X) \wedge has\_motor(X) \wedge \neg$ ( $((has\_wheels(X) \wedge has\_motor(X) \wedge has\_siren(X)) \wedge \neg($ $(has\_siren(X) \wedge off\_duty(X)) \vee (toy(X))$ )) $\vee$ $(toy(X))$ $\vee$ $(has\_wheels(X) \wedge has\_motor(X)$ $\wedge for\_emergencies(X))$ )

PROPOSITION **2.2** Let $\langle\langle A, h\rangle\rangle$ be an argument schema supporting $\langle A', h(\overline{C})\rangle$. If $\langle\langle A', h(\overline{C})\rangle\rangle$ is not a justification, then $f(\mathcal{T}_{\langle A', h(\overline{C})\rangle})$ does not hold. □

However, the converse is not true, since the activation formula does not account for the consistency of the arguments involved. The following algorithm allows us to determine if $\langle\langle A, h(\overline{C})\rangle\rangle$ supports a justification $\langle A', h(\overline{C})\rangle$.

**Algorithm** Justification
**Input:** an abstract dialectical tree $\mathcal{T}_{\langle A, h(\overline{X})\rangle}$, and a vector of constants $\overline{C}$.
**Output:** TRUE if there exists a justification $\langle A', h(\overline{C})\rangle$ supported by $\langle\langle A, h(\overline{X})\rangle\rangle$.

[7]Note that this condition is not sufficient for supporting an argument structure, since $\mathcal{K} \cup A$ might derive $\perp$ (see condition 2 in def. A.2)

**If** $\mathcal{T}_{\langle\langle A, h(\overline{X})\rangle\rangle}$ has a single node
**then**
**If** $\langle\langle A, h(\overline{X})\rangle\rangle$ supports an argument $A'$ for $\overline{C}$
**then** Justification = TRUE
**else** Justification = FALSE
**else**
**If** $\langle\langle A, h(\overline{X})\rangle\rangle$ supports an argument $A'$ for $\overline{C}$
**then**
Let $\mathcal{T}_{\langle\langle A_1, h(\overline{X_1})\rangle\rangle}$, $\mathcal{T}_{\langle\langle A_2, h(\overline{X_2})\rangle\rangle}$, ..., $\mathcal{T}_{\langle\langle A_k, h(\overline{X_k})\rangle\rangle}$ be the immediate subtrees of $\mathcal{T}_{\langle\langle A, h(\overline{X})\rangle\rangle}$.
Let $p_i(\overline{C_i})$ be the counterargumentation literal in $\langle A', h(\overline{C})\rangle$.
Justification = $\neg$ (Justification($\mathcal{T}_{\langle\langle A_1, h(\overline{X_1})\rangle\rangle}$,$\overline{C_1}$) $\vee$
Justification($\mathcal{T}_{\langle\langle A_2, h(\overline{X_2})\rangle\rangle}$,$\overline{C_2}$) $\vee$
$\vdots$
Justification($\mathcal{T}_{\langle\langle A_k, h(\overline{X_k})\rangle\rangle}$,$\overline{C_k}$) )
**else**
Justification = FALSE

The output of this algorithm depends on the structure of the abstract dialectical tree $\mathcal{T}_{\langle\langle A, h(\overline{X})\rangle\rangle}$, independently of the current facts in $\mathcal{K}_P$. In fact, $\mathcal{K}_P$ could be revised (adding or deleting information), without affecting $\mathcal{T}_{\langle\langle A, h(\overline{X})\rangle\rangle}$.

EXAMPLE **2.7** Consider the knowledge base in example 2.1. Let $\mathcal{K}_P$= { *has_motor*(**ambulance**), *person*(**gardener**), *has_wheels*(**ambulance**) }.

Then Justification($\mathcal{T}_{\langle\langle A_1, \neg allowed(X)\rangle\rangle}$, **gardener**) = FALSE, but Justification($\mathcal{T}_{\langle\langle A_1, \neg allowed(X)\rangle\rangle}$, **ambulance**) = TRUE. (*i.e.*, there exists no justified belief for not allowing **gardener** to be in the park, but **ambulance** should not be allowed, since it is a vehicle.)

Suppose our intelligent agent $\mathcal{A}$ adds *has_siren*(**ambulance**) to her ground beliefs, resulting in a $\mathcal{K}_P$' = $\mathcal{K}_P \cup$ {*has_siren*(**ambulance**)}. Now Justification($\mathcal{T}_{\langle\langle A_1, \neg allowed(X)\rangle\rangle}$, **ambulance**) = TRUE (**ambulance** is a vehicle for emergencies, so it should be allowed in the park.)

# 3 Conclusions

We introduced the concepts of *argument schema* and *abstract dialectical tree*, extending the definitions which characterize the *MTDR* framework. This allowed us to speed up inference, which is a fundamental issue for making argumentative formalisms suitable for implementing knowledge-based systems oriented to real applications.

The resulting, extended framework leads to significant computational savings when performing defeasible argumentation. An abstract dialectical tree $\mathcal{T}_{\langle\langle A, h(\overline{X})\rangle\rangle}$ has to be built just once, and its construction can be amortized by many queries, since it subsumes all possible dialectical trees for ground instances of $h(\overline{X})$. Thus, defeasible argumentation can be performed independently of the current ground facts in $\mathcal{K}_P$. Determining if a given argument is a justification results in an elegant procedure, which basically demands a

consistency checking of the arguments involved in its associated dialectical tree. Morover, proposition 2.4 allows us to consider from a different viewpoint the problem of computing specificity efficiently [6], which is a major obstacle in scaling up argumentative systems. Argument schemata offer a new setting in which to investigate this issue.

We contend that the new definitions and algorithms presented in this paper provide a powerful conceptual framework, in which devising new strategies for improving argument-based systems becomes much easier.

# A The MTDR framework

In this appendix, we will briefly describe the main concepts of the *MTDR* framework. For a complete description, see [6].

## A.1 Knowledge representation

The knowledge of an intelligent agent $\mathcal{A}$ will be represented using a first-order language $\mathcal{L}$, plus a binary meta-linguistic relation "$\succ\!\!-$", defined on $\mathcal{L}$, between a set of non-ground literals (antecedent) and a nonground literal (consequent) which share variables. The members of this meta-linguistic relation will be called *defeasible rules*. The relation "$\alpha \succ\!\!- \beta$ " is understood as expressing that "reasons to believe in the antecedent $\alpha$ provide reasons to believe in the consequent $\beta$". We will restrict the first-order language $\mathcal{L}$ to a subset involving only Horn clauses.

The set $\mathcal{K}$ will be a consistent subset of $\mathcal{L}$ representing the non-defeasible part of $\mathcal{A}$'s knowledge. $\Delta$ is a finite set of nonground defeasible rules representing information that $\mathcal{A}$ is prepared to take at less than face value. If $A \subseteq \Delta$, we will denote as $A^{\downarrow}$ the set of all ground instances of members of $A$.

The set $\mathcal{K}$ can be partitioned into two subsets: $\mathcal{K}_G$ (*general* knowledge) and $\mathcal{K}_P$ (*particular* or *contingent* knowledge). Sentences in $\mathcal{K}_P$ will be ground literals (*E.g.*: *flies*(**tweety**), *penguin*(**opus**)) which do not appear as consequents of rules in $\mathcal{K}_G$ or $\Delta$, since they represent basic input information sensored by $\mathcal{A}$, from which new information can be inferred. Sentences in $\mathcal{K}_G$ will be material implications having the form $a_1, a_2, \ldots, a_k \rightarrow b$, *e.g.* $penguin(X) \rightarrow bird(X)$. Defeasible rules have the form $a_1, a_2, \ldots, a_k \succ\!\!- b$, *e.g.* $bird(X) \succ\!\!- flies(X)$.

## A.2 Inference

In order to make this paper self-contained, we present next definitions A.1 through A.8, which summarize the notion of inference in *MTDR* (for a complete definition of this framework, see [6]).

DEFINITION **A.1** Let $\Gamma$ be a subset of $\mathcal{K} \cup \Delta^{\downarrow}$. A ground literal $h$ is a *defeasible consequence* of $\Gamma$, abbreviated $\Gamma \mid\!\sim h$, if and only if there exists a finite sequence $B_1, \ldots, B_n$ such that $B_n = h$ and for $1 \leq i < n$, either $B_i \in \Gamma$, or $B_i$ is a direct consequence of the preceding

elements in the sequence by virtue of the application of any inference rule of the first-order theory associated with the language $\mathcal{L}$. Ground instances of the defeasible rules are regarded as material implications for the application of inference rules. We will write $\mathcal{K} \cup A \mid\sim h$ distinguishing the set $A$ of defeasible rules used in the derivation from the set $\mathcal{K}$. □

DEFINITION **A.2** Given a set $\mathcal{K}$, a set $\Delta$ of defeasible rules, and a ground literal $h$ in the language $\mathcal{L}$, we say that a subset $A$ of $\Delta^{\downarrow}$ is an *argument structure* (or just *argument*) for $h$ in the context $\mathcal{K}$ (denoted by $\langle A, h\rangle_{\mathcal{K}}$, or just $\langle A, h\rangle$) if and only if: 1) $\mathcal{K} \cup A \mid\sim h$, 2) $\mathcal{K} \cup A \not\mid\sim \bot$ and 3) $\nexists A' \subset A$, $\mathcal{K} \cup A' \mid\sim h$. A *subargument* of $\langle A, h\rangle$ is an argument $\langle S, j\rangle$ such that $S \subseteq A$. □

DEFINITION **A.3** Given two arguments $\langle A_1, h_1\rangle$ and $\langle A_2, h_2\rangle$, we say that $\langle A_1, h_1\rangle$ *counterargues* $\langle A_2, h_2\rangle$, denoted $\langle A_1, h_1\rangle \overset{h}{\otimes\!\!\rightarrow} \langle A_2, h_2\rangle$ iff there exists a subargument $\langle A, h\rangle$ of $\langle A_2, h_2\rangle$ such that $\mathcal{K} \cup \{h_1, h_2\} \vdash \bot$. The literal $h$ will be called a *counterargumentation literal.* □

DEFINITION **A.4** Let $\mathcal{D} = \{a \in \mathcal{L} : a$ is a ground literal and $\mathcal{K} \cup \Delta^{\downarrow} \mid\sim a\}$, and let $\langle A_1, h_1\rangle$ and $\langle A_2, h_2\rangle$ be two argument structures. We say that $A_1$ for $h_1$ is *strictly more specific than* $A_2$ for $h_2$, denoted $\langle A_1, h_1\rangle \succ_{\mathbf{spec}} \langle A_2, h_2\rangle$, if and only if
i) $\forall S \subseteq \mathcal{D}$ if $\mathcal{K}_G \cup S \cup A_1 \mid\sim h_1$ and $\mathcal{K}_G \cup S \not\mid\sim h_1$, then $\mathcal{K}_G \cup S \cup A_2 \mid\sim h_2$.
ii) $\exists S \subseteq \mathcal{D}$ such that $\mathcal{K}_G \cup S \cup A_2 \mid\sim h_2$, $\mathcal{K}_G \cup S \not\mid\sim h_2$ and $\mathcal{K}_G \cup S \cup A_1 \not\mid\sim h_1$. □

DEFINITION **A.5** Given two argument structures $\langle A_1, h_1\rangle$ and $\langle A_2, h_2\rangle$, we say that $\langle A_1, h_1\rangle$ *defeats* $\langle A_2, h_2\rangle$ at literal $h$, denoted $\langle A_1, h_1\rangle \gg_{\mathbf{def}} \langle A_2, h_2\rangle$, if and only if there exists a subargument $\langle A, h\rangle$ of $\langle A_2, h_2\rangle$ such that: $\langle A_1, h_1\rangle$ counterargues $\langle A_2, h_2\rangle$ at the literal $h$ and
1. $\langle A_1, h_1\rangle$ is *strictly more specific*[8] than $\langle A, h\rangle$, or 2. $\langle A_1, h_1\rangle$ is unrelated by specificity to $\langle A, h\rangle$.
If $\langle A_1, h_1\rangle \gg_{\mathbf{def}} \langle A_2, h_2\rangle$, we will also say that $\langle A_1, h_1\rangle$ is a *defeater* for $\langle A_2, h_2\rangle$. In case (1) $\langle A_1, h_1\rangle$ will be called a *proper* defeater, and in case (2) a *blocking* defeater. □

DEFINITION **A.6** A *dialectical tree* $\mathcal{T}_{\langle A, h\rangle}$ for an argument $\langle A, h\rangle$ is recursively defined as follows:

1. A single node containing an argument structure $\langle A, h\rangle$ with no defeaters is by itself a dialectical tree for $\langle A, h\rangle$. This node is also the root of the tree.

2. Suppose that $\langle A, h\rangle$ is an argument structure with defeaters $\langle A_1, h_1\rangle$, $\langle A_2, h_2\rangle$, ..., $\langle A_n, h_n\rangle$. We construct the dialectical tree $\mathcal{T}_{\langle A, h\rangle}$, by putting $\langle A, h\rangle$ as the root node of $\mathcal{T}_{\langle A, h\rangle}$ and by making this node the parent node of the roots of the dialectical trees for $\langle A_1, h_1\rangle$, $\langle A_2, h_2\rangle$, ..., $\langle A_n, h_n\rangle$.

□

[8]We use specificity as comparison criterion, but any other partial order among arguments might be possible.

DEFINITION **A.7** Let $\mathcal{T}_{\langle A,h\rangle}$ be a dialectical tree for an argument structure $\langle A,h\rangle$. The nodes of $\mathcal{T}_{\langle A,h\rangle}$ can be recursively labeled as *undefeated nodes* (U-nodes) and *defeated nodes* (D-nodes) as follows:

1. Leaves of $\mathcal{T}_{\langle A,h\rangle}$ are *U-nodes.*
2. Let $\langle B,q\rangle$ be an inner node of $\mathcal{T}_{\langle A,h\rangle}$. Then $\langle B,q\rangle$ will be an *U-node* iff every child of $\langle B,q\rangle$ is a *D-node.* $\langle B,q\rangle$ will be a *D-node* iff it has at least an *U-node* as a child.

□

DEFINITION **A.8** Let $\langle A,h\rangle$ be an argument structure, and let $\mathcal{T}_{\langle A,h\rangle}$ be a dialectical tree. [9] We will say that $A$ is a *justification* for $h$ (or simply $\langle A,h\rangle$ is a *justification*) iff the root node of $\mathcal{T}_{\langle A,h\rangle}$ is an U-node. □

# References

[1] A. J. García, C. I. Chesñevar, and G. R. Simari. Making Argument Systems Computationally Attractive. In *Anales de la XIII Conferencia Internacional de la Sociedad Chilena para Ciencias de la Computación*, Universidad de La Serena, La Serena (Chile), 1993.

[2] A. Y. Levy. *Irrelevance Reasoning in Knowledge Based Systems.* PhD thesis, Stanford University, Departament of Computer Science, Oct. 1993.

[3] R. Loui and J. Norman. *Rationales and Argument Moves.* Technical Report, Washington University, Dept. of Computer Science, St.Louis, USA, Oct. 1992.

[4] G. R. Simari, C. I. Chesñevar, and A. J. García. Focusing inference in defeasible argumentation. In *Anales de la Conferencia IBERAMIA '94*, Asociación Venezolana para Inteligencia Artificial, Caracas (Venezuela), Oct. 1994.

[5] G. R. Simari, C. I. Chesñevar, and A. J. García. The role of dialectics in defeasible argumentation. In *Anales de la XIV Conferencia Internacional de la Sociedad Chilena para Ciencias de la Computación*, Universidad de Concepción, Concepción (Chile), Nov. 1994.

[6] G. R. Simari and R. P. Loui. A Mathematical Treatment of Defeasible Reasoning and its Implementation. *Artificial Intelligence*, 53:125–157, 1992.

[9] Actually, dialectical trees should satisfy certain additional requirements for being considered *acceptable* dialectical trees (see [5]). These requirements do not affect our discussion about argument schemata, and may be easily introduced into the resulting framework. This issue, however, exceeds the scope of this paper.