A Mathematical Conceptualization of Bundle Sets in Defeasible Logic Programming

Yamil O. Soto
Dept. of CS and Eng., Universidad Nacional del Sur (UNS)
& Inst. for CS and Eng. (ICIC UNS-CONICET)
Bahia Blanca, Buenos Aires, Argentina
yamil.soto@cs.uns.edu.ar

Cristhian Ariel D. Deagustini
Area de Agentes y Sistemas Inteligentes (FCAD UNER)
Concordia, Entre Rios, Argentina
ariel.deagustini@uner.edu.ar

Maria Vanina Martinez
Artificial Intelligence Research Institute (IIIA-CSIC)
Barcelona, Catalonia, Spain
vmartinez@iiia.csic.es

Gerardo I. Simari
Dept. of CS and Eng., Universidad Nacional del Sur (UNS)
& Inst. for CS and Eng. (ICIC UNS-CONICET)
Bahia Blanca, Buenos Aires, Argentina
gis@cs.uns.edu.ar

ABSTRACT

Defeasible Logic Programming (DeLP) is a formalism for structured argumentation-based reasoning that is founded on a dialectical procedure that relies on trees to compute answers to queries, which return the so-called warrant statuses of the literals involved. In this work, we propose a novel, more general, and minimal structure (understanding it as the minimum information necessary to warrant a statement) than the concept of dialectical tree, which we call *parsimonious bundle set*. This structure is significant for a line of research that we are carrying out in which we are studying the theoretical foundations of DeLP toward the definition of model-theoretic semantics, a contribution that, in turn, will allow us to analyze the formalism from a different perspective, focusing on theoretical aspects, and also to eventually generalize the query language from literals to formulas built from Boolean connectives. Therefore, in this first step, we focus on developing the basic tools required to provide an alternative to the operational semantics of DeLP that is centered on declarative definitions.

CCS CONCEPTS

• **Computing methodologies → Nonmonotonic, default reasoning and belief revision**.

KEYWORDS

Defeasible reasoning, Structured Argumentation, Defeasible Logic Programming, Argumentation lines, Model-theoretic semantics

ACM Reference Format:
Yamil O. Soto, Cristhian Ariel D. Deagustini, Maria Vanina Martinez, and Gerardo I. Simari. 2024. A Mathematical Conceptualization of Bundle Sets in Defeasible Logic Programming. In *The 39th ACM/SIGAPP Symposium on Applied Computing (SAC '24), April 8–12, 2024, Avila, Spain.* ACM, New York, NY, USA, 8 pages. https://doi.org/10.1145/3605098.3636042

SAC '24, April 8–12, 2024, Avila, Spain

ACM ISBN 979-8-4007-0243-3/24/04…$15.00
https://doi.org/10.1145/3605098.3636042

1 INTRODUCTION AND RELATED WORK

Defeasible Logic Programming (DeLP) [8] is a structured argumentative approach to defeasible reasoning [11, 17], *i.e.*, conflicting logical arguments supporting specific queries are considered in a global process to determine which survive. In DeLP, queries are analyzed in a *dialectical process* that exhaustively considers arguments for and against specific answers in search of *warrants*, which means that available information supports specific conclusions. In this way, a query α has answer *Yes* when there is an argument for α that is warranted, *No* when there is an argument for $\sim\alpha$ that is warranted, *Undecided* if neither α nor $\sim\alpha$ have arguments that warrant them, and *Unknown* otherwise. DeLP is an interesting formalism since it allows to deal with both incomplete and contradictory information in dynamic domains; hence, the framework is suitable for representing agents' knowledge and providing them with an argumentation-based reasoning mechanism, and it has proven to be successful in real-world applications [4, 7, 12, 16].

Presumptive Defeasible Logic Programming (PreDeLP) [14] is an extension of DeLP in which presumptions are fully integrated into the reasoning mechanism. A presumption is a piece of information that is tentatively taken to be true although it is not known for certain, usually in the absence of acceptable reasons to the contrary. Though syntactically presumptions look like defeasible facts, treating them adequately requires a more involved approach.

Formal semantics have been defined for Abstract Argumentation Frameworks [1] in terms of models (extensions), much like in Logic Programming [10], defining the subset of all models that satisfy some desirable properties. To date, no model-theoretic semantics has been developed neither for DeLP nor PreDeLP; a promising line of research, that we propose as future work, is to develop such semantics with the ultimate goal of studying the theoretical foundations of PreDeLP. As an additional benefit, this may also pave the way towards generalizing the query language, and it may also help programmers better understand program behavior and results.

In this paper, we propose a novel structure, called *parsimonious bundle set*, which we believe will be useful in developing the above-mentioned line of research. Parsimonious bundle sets are more

general structures than dialectical trees, defined in a more declarative way, and which are also *minimal* (understood as containing the minimal information, according to set inclusion, necessary to warrant a claim). In this first work, we define a set of operations that can be performed over these structures and present several interesting properties they enjoy. The main contributions of this work are: (i) We capture the notion of minimality of information needed to warrant a claim through the notion of *parsimony*; (ii) we propose a structure, called parsimonious bundle sets, that is minimal and more general than dialectical trees, while still encoding all the information needed to determine the warrant status of a root argument; (iii) we explore how several set theory-inspired operations on bundle sets can be defined; and (iv) we define a warrant procedure for query answering based on parsimonious bundle sets.

The rest of the paper is organized as follows. In Section 2 we briefly recall the language and the main concepts of PreDeLP. In Section 3.1 we recall some notions about bundle sets, and in Section 3.2 we characterize the notion of parsimonious bundle set, give operations on them, and define a warrant procedure through this structure. Finally, we discuss the main conclusions of the paper and some future lines of research in Section 4.

2 DEFEASIBLE LOGIC PROGRAMMING WITH PRESUMPTIONS

In the following, we briefly recall the most relevant concepts for PreDeLP from [14]. In the PreDeLP language, a literal L is a (possibly negated) ground atom. We represent strong negation with "$\sim$" and say that L and $\sim L$ are complementary. A *strict* (resp., *defeasible*) *rule* has the form $L \leftarrow L_1, \ldots, L_n$ (resp., $L \prec L_1, \ldots, L_n$), where L is the *head*, and $L_1, \ldots, L_n$, with $n \geqslant 1$, is the *body* of the rule. $Body(R)$ and $Head(R)$ denote the body and head of a rule R. A defeasible rule $head \prec body$ represents a weaker connection between *body* and *head*. It can be understood as expressing that "reasons to believe in the antecedent *body* provide reasons to believe in the consequent *head*" [21], but it may be the case that *body* is true and *head* is not. Note that the symbols $\leftarrow$ and $\prec$ denote meta-relations between a literal and a set of literals, and have no interaction with language symbols. As in Logic Programming, strict and defeasible rules are not conditionals nor implications, they are inference rules [8, 9]. A *fact* (resp., *presumption*) is a strict (resp., defeasible) rule with an empty body, denoted as L (resp., $L \prec$). Intuitively, presumptions are pieces of information that are tentatively taken to be true, usually in the absence of acceptable reasons to the contrary. They express motives to believe in some information, and they represent weaker assertions than facts.

Definition 2.1. A *PreDeLP program* $P = (\Omega, \Theta, \Delta, \Phi)$, is a set of strict rules Ω, facts Θ, defeasible rules Δ, and presumptions Φ.

Next, we introduce the concept of *annotated derivation*, where we subscribe to the definition given in [14]. In what follows, we use the term "annotated derivation" or just "derivation".

Definition 2.2. Let P be a PreDeLP program and L a literal. An *annotated derivation*, denoted ∂, of L from P consists of a finite sequence of rules, facts, and possibly presumptions $[R_1, \ldots, R_n]$, where L is (i) a fact R_n, (ii) a presumption R_n, or (iii) the head of the rule R_n. Furthermore, if a rule R_i is in the sequence, then its body $B_1, \ldots, B_k$, is such that for all B_j, with $1 \leqslant j \leqslant k$, B_j is a fact, a presumption or it appears as the head L_m, for some rule R_m with $1 \leqslant m < i$.

We assume a canonical form for derivations. A derivation $\partial' = [R_1, \ldots, R_i]$ is a sub-derivation of $\partial = [R_1, \ldots, R_n]$ if $i \leqslant n$. A derivation ∂ is *strict* when neither presumptions nor defeasible rules are used in ∂, otherwise ∂ is *defeasible*. A literal L is *strictly derived* from P, denoted $P \vdash L$, if there exists a strict derivation for L from P, and L is *defeasibly derived* from P, denoted $P \mid\sim L$, if there exists a defeasible derivation for L from P and no strict derivation exists. A derivation ∂ for L is *minimal* if no proper sub-derivation ∂' of ∂ is also a derivation of L. Considering minimal derivations avoids the insertion of unnecessary elements that will weaken its ability to support the conclusion by possibly introducing needless points of conflict. Given a derivation ∂ for literal L, there always exists at least one minimal sub-derivation ∂' for L. A PreDeLP program $P = (\Omega, \Theta, \Delta, \Phi)$ is contradictory if there exist derivations for two complementary literals. We denote with $\Pi = (\Omega, \Theta)$ the *strict part* of P, and assume that the sub-program Π is *non-contradictory*. Two literals L_1 and L_2 *disagree* w.r.t. P if $\Pi \cup \{L_1, L_2\}$ is contradictory. Intuitively, this means that both should not be inferred and a mechanism to decide between one or the other is needed. This is done through an argumentative approach that consists of building arguments for the literals in conflict—we call such literals *claims*—and evaluating all such arguments in a dialectical process to decide which prevails. A derivation ∂ is *contradictory* if two disagreeing literals L_1 and L_2 can be derived from it. In the following, we only consider minimal and non-contradictory derivations.

2.1 Constructing and Comparing Arguments

Intuitively, arguments are structures that support a claim from evidence through a reasoning mechanism.

Definition 2.3. An *argument* for a literal α from a PreDeLP program P, denoted $\langle \mathcal{A}, \alpha \rangle$, is the set $\mathcal{A}$ of facts, presumptions, and rules (strict and defeasible) used in an annotated derivation ∂ for α. An argument $\mathcal{A}$ is a *sub-argument* of an argument $\mathcal{B}$ if $\mathcal{A} \subseteq \mathcal{B}$.

$\mathcal{A}$ is called the set of premises and α is called the conclusion or claim. Since we only consider minimal and non-contradictory annotated derivations, the set of premises $\mathcal{A}$ used in ∂ is minimal and non-contradictory. Non-minimal or contradictory sets of premises are never arguments. An argument $\langle \mathcal{A}, \alpha \rangle$ is *strict* iff α is strictly derived from $\mathcal{A}$, otherwise $\langle \mathcal{A}, \alpha \rangle$ is a *defeasible argument*. In the following, we will refer to an argument $\langle \mathcal{A}, \alpha \rangle$ simply as $\mathcal{A}$ when its claim is not relevant to the discussion. Answers to queries are supported by arguments built from the program. It is possible to build arguments for complementary literals, and arguments can thus *attack* each other.

Definition 2.4. Let $\langle \mathcal{A}, \alpha \rangle$ and $\langle \mathcal{B}, \beta \rangle$ be two arguments in a PreDeLP program P. $\langle \mathcal{A}, \alpha \rangle$ *attacks* $\langle \mathcal{B}, \beta \rangle$ at literal γ, iff there exists a sub-argument $\langle C, \gamma \rangle$ of $\langle \mathcal{B}, \beta \rangle$ such that α and γ disagree.

Given argument $\mathcal{A}$ and counter-argument $\mathcal{B}$, a *comparison criterion* is used to decide if $\mathcal{A}$ is preferred to $\mathcal{B}$ and, therefore, defeats $\mathcal{B}$. The definition of such a formal criterion is a central problem in any argumentation system where the defeat relation must be computed from the structure of arguments. Although the comparison

criterion used in DeLP, and by extension in PreDeLP, is modular, *Generalized Specificity* [22] is the default. This criterion intuitively favors arguments with greater information content (classical Specificity) or with less use of rules (a more direct derivation). However, in the presence of presumptions, Generalized Specificity does not always have the intended results; for that reason, other preference criteria have been developed [14]. In this work, we assume an arbitrary comparison criterion denoted with $\mathcal{A} \succ \mathcal{B}$ whenever $\mathcal{A}$ is preferred to $\mathcal{B}$.

2.2 Dialectical Analysis

The dialectical process exhaustively considers arguments for or against specific queries looking for a *warrant*, which means that the claim is supported by an undefeated argument. To decide whether an argument is undefeated within a program, all related arguments, *i.e.*, arguments that support or interfere with it, must be considered.

Definition 2.5. Let $\mathcal{A}$ and $\mathcal{B}$ be arguments from a PreDeLP program P such that $\mathcal{A}$ attacks $\mathcal{B}$, and $\succ$ is a comparison criterion.

- $\mathcal{A}$ is a *proper defeater* of $\mathcal{B}$ iff $\mathcal{A} \succ \mathcal{B}$.
- $\mathcal{A}$ is a *blocking defeater* of $\mathcal{B}$ iff $\mathcal{A} \not\succ \mathcal{B}$ and $\mathcal{B} \not\succ \mathcal{A}$.
- $\mathcal{A}$ *defeats* $\mathcal{B}$, denoted $(\mathcal{A}, \mathcal{B})$, iff $\mathcal{A}$ is a proper or a blocking defeater of $\mathcal{B}$.

Definition 2.6. Let $\mathcal{A}_1$ be an argument in a PreDeLP program P. An *argumentation line* for or rooted in $\mathcal{A}_1$ is a sequence of arguments of the form $\Lambda = [\mathcal{A}_1, \ldots, \mathcal{A}_n]$ where each element is a (blocking or proper) defeater of its predecessor.

Different argumentation systems can be defined by setting a particular criterion for proper attacks or defining the admissibility of argumentation lines. Here, we adopt the one from [8].

Definition 2.7. Given an argumentation line Λ:

- The *supporting set* is $S(\Lambda) = \{\mathcal{A}_i \mid \mathcal{A}_i \in \Lambda \text{ and } i = 2k+1\}$ with $k \in \mathbb{N}$, *i.e.*, i is odd.
- The *interfering set* is $I(\Lambda) = \{\mathcal{A}_i \mid \mathcal{A}_i \in \Lambda \text{ and } i = 2k\}$ with $k \in \mathbb{N}_{>0}$, *i.e.*, i is even.

Definition 2.8. Let $(\Omega, \Theta, \Delta, \Phi)$ be a PreDeLP program. Two arguments $\langle \mathcal{A}, \alpha \rangle$ and $\langle \mathcal{B}, \beta \rangle$ are *concordant* iff the set $\Pi \cup \mathcal{A} \cup \mathcal{B}$ is non-contradictory. More generally, a set of arguments $\{\langle \mathcal{A}_i, \alpha_i \rangle\}_{i=1}^{n}$ is concordant iff $\Pi \cup \bigcup_{i=1}^{n} \mathcal{A}_i$ is non-contradictory.

Definition 2.9. An argumentation line Λ is *acceptable* if:

- Λ is finite.
- The supporting (resp., interfering) set is concordant.
- No argument $\mathcal{A}_i$ in Λ is a sub-argument of an argument $\mathcal{A}_h$ appearing earlier in Λ (with $h < i$).
- For all i, such that the argument $\mathcal{A}_i$ is a blocking defeater for $\mathcal{A}_{i-1}$, if $\mathcal{A}_{i+1}$ exists, then $\mathcal{A}_{i+1}$ is a proper defeater for $\mathcal{A}_i$.

The dialectical process considers all possible acceptable argumentation lines for an argument, which together form a *dialectical tree*. Such trees for PreDeLP programs are defined following [8], and we adopt the notion of *coherent dialectical tree* from [14], which, intuitively, ensures that conflicting presumptions are not used together in supporting (or attacking) a claim.

Definition 2.10. Let $\mathcal{A}_1$ be an argument from a PreDeLP program P. A *dialectical tree* for $\mathcal{A}_1$, denoted $\mathcal{T}(\mathcal{A}_1)$, is defined as follows:

- The root of the tree is labeled with $\mathcal{A}_1$.
- Let N be a node of the tree labeled $\mathcal{A}_n$, and $\Lambda = [\mathcal{A}_1, \ldots, \mathcal{A}_n]$ be the sequence of labels of the path from the root to N. Let $\{\mathcal{B}_1, \ldots, \mathcal{B}_k\}$ be all the defeaters for $\mathcal{A}_n$ from P. For each defeater $\mathcal{B}_i$ $(1 \leqslant i \leqslant k)$, such that $\Lambda' = [\mathcal{A}_1, \ldots, \mathcal{A}_n, \mathcal{B}_i]$ is an acceptable argumentation line, the node N has a child N_i labeled $\mathcal{B}_i$. If there is no defeater for $\mathcal{A}_n$ or there is no $\mathcal{B}_i$ such that Λ' is acceptable, then N is a leaf.

Argument evaluation, *i.e.*, determining whether the root node of a dialectical tree is defeated or undefeated, is done through a *marking* or *labeling* procedure, according to which each node in a dialectical tree is labeled as either defeated (D) or undefeated (U). Let $root(\mathcal{T}(\mathcal{A}))$ be the root of $\mathcal{T}(\mathcal{A})$ and $mark(N)$ the value of the marking for node N in $\mathcal{T}(\mathcal{A})$.

Definition 2.11. Let $\mathcal{A}$ be an argument in a PreDeLP program P. Let $\mathcal{T}(\mathcal{A})$ be a dialectical tree for $\mathcal{A}$. The *marking* of $\mathcal{T}(\mathcal{A})$ will be obtained marking every node in $\mathcal{T}(\mathcal{A})$ as follows:

- All leaves in $\mathcal{T}(\mathcal{A})$ are marked as U.
- Let $\mathcal{B}$ be an inner node of $\mathcal{T}(\mathcal{A})$. Then $\mathcal{B}$ is marked as U iff every child of $\mathcal{B}$ is marked as D. Node $\mathcal{B}$ is marked as D iff it has at least one child that is marked as U.

This dialectical process allows us to define a *query answering semantics*, according to which a literal α is *warranted* in a PreDeLP program P if there exists a tree whose root is an argument for α marked as undefeated. This semantics enjoys the so-called *direct consistency* property [3], *i.e.*, no contradictory sets of literals can be warranted from P.

Definition 2.12. A literal α is *warranted* from a PreDeLP program P, denoted $P \mid\sim_W \alpha$, iff there exists a dialectical tree $\mathcal{T}(\mathcal{A})$ rooted in $\langle \mathcal{A}, \alpha \rangle$ such that $mark(root(\mathcal{T}(\mathcal{A}))) = U$. If $\sim\alpha$ is warranted, then α is not warranted. If neither α nor $\sim\alpha$ are warranted, then α is *Undecided*. If α is not part of the language of P, it is *Unknown*.

3 TOWARDS WARRANTING BY MEANS OF PARSIMONIOUS BUNDLE SETS

In Section 3.1 we recall some basic notions presented in [5, 6] and define a marking and warrant procedure on bundle sets; then, in Section 3.2 we substantially extend the presented framework to capture the notion of a more general and minimal structure than dialectical trees, while still encoding the necessary information to determine the warrant status of a root argument.

3.1 Introducing Bundle Sets

In the following, we will refer to finite argumentation lines but not necessarily acceptable ones; also, for simplicity, we use the term "argumentation lines", or just "lines".

Definition 3.1. Let $\Lambda = [\mathcal{A}_1, \ldots, \mathcal{A}_n]$ be an argumentation line. A *segment* for Λ is an initial sequence $\Lambda' = [\mathcal{A}_1, \ldots, \mathcal{A}_i]$ with $i \leqslant n$. A segment $\Lambda' = [\mathcal{A}_1, \ldots, \mathcal{A}_i]$ for Λ is *proper* if $i < n$.

Remark 1. *A segment Λ of an argumentation line Λ'' is always a line and also the segment of another segment Λ' of Λ''.*

Example 1. *Consider a PreDeLP program P where the following set of arguments can be built: $\{\mathcal{A}_1, \mathcal{A}_2, \mathcal{A}_3, \mathcal{A}_4\}$, and where (only)*

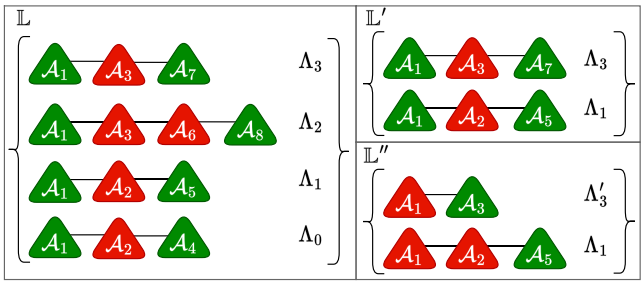

Figure 1: $\mathbb{L}$, $\mathbb{L}'$, and $\mathbb{L}''$ are the exhaustive, a parsimonious, and a partial bundle set, respectively, for $\mathcal{A}_1$. Undefeated (resp., defeated) arguments are colored green (resp., red).

the following defeat relations hold: $(\mathcal{A}_2, \mathcal{A}_1)$, $(\mathcal{A}_3, \mathcal{A}_2)$, $(\mathcal{A}_4, \mathcal{A}_1)$, *and* $(\mathcal{A}_1, \mathcal{A}_4)$. *Note that the last two relations represent that* $\mathcal{A}_1$ *and* $\mathcal{A}_4$ *are blocking defeaters. Three possible argumentation lines rooted in* $\mathcal{A}_1$ *are* $\Lambda_1 = [\mathcal{A}_1]$, $\Lambda_2 = [\mathcal{A}_1, \mathcal{A}_2]$, *and* $\Lambda_3 = [\mathcal{A}_1, \mathcal{A}_2, \mathcal{A}_3]$, *where* Λ_1 *is a proper segment of* Λ_2 *and* Λ_2 *is a proper segment of* Λ_3. *Also, an infinite number of lines rooted in* $\mathcal{A}_1$ *can be obtained through the relation between* $\mathcal{A}_1$ *and* $\mathcal{A}_4$, e.g., $\Lambda_4 = [\mathcal{A}_1, \mathcal{A}_4]$, $\Lambda_5 = [\mathcal{A}_1, \mathcal{A}_4, \mathcal{A}_1]$, $\Lambda_6 = [\mathcal{A}_1, \mathcal{A}_4, \mathcal{A}_1, \mathcal{A}_4]$, *etc. Note that in the previous case, each* Λ_i *is a proper segment of* Λ_j *with* $4 \leqslant i < j$.

An argumentation line Λ is *exhaustive* if there is no line Λ' such that Λ is a proper segment of Λ'. Otherwise, Λ is *partial*. In Example 1, argumentation line Λ_3 is exhaustive, while all the others are partial. A *bundle set* is a set of argumentation lines such that no line in the set is a proper segment of another line in the set.

Definition 3.2. Let $\mathbb{L}$ be a set of argumentation lines rooted in argument $\mathcal{A}$ from a PreDeLP program P. $\mathbb{L}$ is a *bundle set* for $\mathcal{A}$ if there is no $\Lambda, \Lambda' \in \mathbb{L}$ such that Λ is a proper segment of Λ'.

We are interested in bundle sets of acceptable argumentation lines, which we call *acceptable bundle sets*. From now on, all bundle sets that we consider are acceptable unless otherwise stated. The concept of bundle set is a more general one than that of dialectical tree; intuitively, if we consider a bundle set rooted in argument $\mathcal{A}$, we can construct a tree structure for $\mathcal{A}$ where each argumentation line is a branch in the tree structure. Indeed, several equivalent tree structures can be constructed from the same bundle set (see Definitions 3.5, 3.6, and 3.7 for further details). Also, the notion of bundle set allows us to treat dialectical trees as sets whose elements are argumentation lines, and then operate with them analogously to operations in set theory (see Section 3.2). Given the above, from now on we will focus on bundle sets. All the properties that we highlight for bundle sets also extend to tree structures.

The following definition formalizes two specific kinds of bundle sets that we are interested in.

Definition 3.3. Let $\mathbb{L}$ be a bundle set rooted in argument $\mathcal{A}$ from a PreDeLP Program P. $\mathbb{L}$ is *exhaustive* iff $\mathbb{L}$ is the set of all exhaustive lines rooted in $\mathcal{A}$, *i.e.*, there exists no Λ'' rooted in $\mathcal{A}$ such that:

1. $\Lambda'' \notin \mathbb{L}$,
2. Λ' is a proper segment of Λ'', where Λ' is a segment of some $\Lambda \in \mathbb{L}$, and
3. Λ'' is not a proper segment of any $\Lambda \in \mathbb{L}$.

Otherwise, $\mathbb{L}$ is *partial*.

Intuitively, if there exists an argumentation line Λ'' rooted in $\mathcal{A}$ that does not belong to $\mathbb{L}$ (condition 1) such that Λ'' has a proper segment Λ' which is a segment of some line in $\mathbb{L}$ (condition 2) and Λ'' is not a proper segment of any line Λ in $\mathbb{L}$ (condition 3), then one of the lines in $\mathbb{L}$ is not exhaustive or Λ'' is a new line not in $\mathbb{L}$. In both cases, $\mathbb{L}$ is not the set of all exhaustive argumentation lines rooted in $\mathcal{A}$; consequently, it is partial.

Note that there exists only one exhaustive bundle set for any given argument. Also, note that dialectical trees are exhaustive bundle sets, *i.e.*, consider all defeaters for a given argument, all the defeaters of the defeaters, and so on.

Example 2. *Let* P *be a PreDeLP program where the following set of arguments can be built:* $\{\mathcal{A}_1, \mathcal{A}_2, \mathcal{A}_3, \mathcal{A}_4, \mathcal{A}_5, \mathcal{A}_6, \mathcal{A}_7, \mathcal{A}_8\}$, *and where (only) the following defeat relations hold:* $(\mathcal{A}_2, \mathcal{A}_1)$, $(\mathcal{A}_3, \mathcal{A}_1)$, $(\mathcal{A}_4, \mathcal{A}_2)$, $(\mathcal{A}_5, \mathcal{A}_2)$, $(\mathcal{A}_6, \mathcal{A}_3)$, $(\mathcal{A}_7, \mathcal{A}_3)$, $(\mathcal{A}_8, \mathcal{A}_6)$. *The possible exhaustive argumentation lines rooted in* $\mathcal{A}_1$ *are:* $\Lambda_0 = [\mathcal{A}_1, \mathcal{A}_2, \mathcal{A}_4]$, $\Lambda_1 = [\mathcal{A}_1, \mathcal{A}_2, \mathcal{A}_5]$, $\Lambda_2 = [\mathcal{A}_1, \mathcal{A}_3, \mathcal{A}_6, \mathcal{A}_8]$ *and* $\Lambda_3 = [\mathcal{A}_1, \mathcal{A}_3, \mathcal{A}_7]$. *The bundle set* $\mathbb{L} = \{\Lambda_0, \Lambda_1, \Lambda_2, \Lambda_3\}$ *is exhaustive because it is the set of all exhaustive lines rooted in* $\mathcal{A}_1$. *The bundle set* $\mathbb{L}'' = \{\Lambda_1, \Lambda'_3\}$, *where* $\Lambda'_3 = [\mathcal{A}_1, \mathcal{A}_3]$, *is a partial one because one of its lines is partial* (Λ'_3 *is a proper segment of both,* Λ_2 *and* Λ_3), *and also because although* Λ_1 *is exhaustive, (at least) one of its segments* $[\mathcal{A}_1, \mathcal{A}_2]$ *is a proper segment of* Λ_0. *Figure 1 shows these examples.*

We are now almost ready to reformulate the notion of a warranted literal based on bundle sets; first, we need to extend some concepts to formalize this idea.

Definition 3.4. Let $\mathbb{L}$ be a bundle set for argument $\mathcal{A}_1$ from a PreDeLP program P. The *marking* of $\mathbb{L}$ will be obtained by marking each occurrence of an argument in some $\Lambda \in \mathbb{L}$ as follows:

1. Let $\Lambda = [\mathcal{A}_1, \ldots, \mathcal{A}_n] \in \mathbb{L}$; then, $\mathcal{A}_n$ is marked as U in Λ.
2. Let $\Lambda = [\mathcal{A}_1, \ldots, \mathcal{A}_i, \ldots, \mathcal{A}_n] \in \mathbb{L}$, with $i < n$, and let $E = \{\Lambda' \in \mathbb{L} \mid [\mathcal{A}_1, \ldots, \mathcal{A}_i]$ be a proper segment of $\Lambda'\}$. $\mathcal{A}_i$ is marked as U in Λ iff for every line $\Lambda' \in E$ it holds that the defeater of $\mathcal{A}_i$ is marked as D in Λ'. Otherwise, $\mathcal{A}_i$ is marked as D.

Intuitively, condition 1 states that if an argument is the last element of a line Λ, it does not have any (valid w.r.t. Definition 2.9) defeater and, consequently, is marked as U in Λ. Condition 2 says that if an argument $\mathcal{A}_i$ is an "internal" element of a set of lines E, then the mark of all its defeaters in all lines in E must be considered, and $\mathcal{A}_i$ is marked in the same way in all lines belonging to E.

With $root(\mathbb{L})$ we denote the argument in which $\mathbb{L}$ is rooted, and the marking of the occurrence of some argument $\mathcal{A}_i \in \Lambda \in \mathbb{L}$ is denoted with $mark(\mathcal{A}_i, \Lambda)$. Figure 1 shows examples of markings for different bundle sets. For instance, in $\mathbb{L}'$, $\mathcal{A}_1$ is marked as U in Λ_1 and Λ_3 because all its defeaters in all lines in the bundle set, namely $\mathcal{A}_2 \in \Lambda_1$ and $\mathcal{A}_3 \in \Lambda_3$, are defeated. On the other hand, in $\mathbb{L}''$, $\mathcal{A}_1$ is marked as D in Λ_1 and Λ'_3 because although there is a defeater of $\mathcal{A}_1$ that is defeated, namely $\mathcal{A}_2 \in \Lambda_1$, there is also another defeater of $\mathcal{A}_1$, that is $\mathcal{A}_3 \in \Lambda'_3$, which is undefeated.

We now propose a constructive definition of a more general concept of a tree structure than a dialectical tree, which we call argumentation tree, from a bundle set. With some abuse of notation, we denote argumentation trees in the same way as dialectical ones.

Definition 3.5. Let $\mathbb{L}$ be a bundle set for argument $\mathcal{A}$ in a PreDeLP program P. An *argumentation tree* $\mathcal{T}(\mathcal{A})$, can be built from $\mathbb{L}$ in the following manner:

- The root of $\mathcal{T}(\mathcal{A})$ is labeled with $\mathcal{A}$.
- Let $F = \{tail(\Lambda)$, for all $\Lambda \in \mathbb{L}\}$, and $H = \{head(\Lambda)$, for all $\Lambda \in F\}$[1]. If $H = \emptyset$ then $\mathcal{T}(\mathcal{A})$ has no sub-trees. Otherwise, if $H = \{\mathcal{B}_1, \ldots, \mathcal{B}_k\}$, then for every $\mathcal{B}_i \in H$, we define $getBundle(\mathcal{B}_i) = \{\Lambda \in F \mid head(\Lambda) = \mathcal{B}_i\}$. We put $\mathcal{T}(\mathcal{B}_i)$ as an immediate sub-tree of $\mathcal{T}(\mathcal{A})$, where $\mathcal{T}(\mathcal{B}_i)$ is an argumentation tree based on $getBundle(\mathcal{B}_i)$.

Note that dialectical trees are a particular case of argumentation trees. Intuitively, the main difference between both kinds of trees is that argumentation ones are not necessarily exhaustive, *i.e.*, they do not necessarily consider all the defeaters (that do not violate any of the restrictions of Definition 2.9) for an argument in the tree, while dialectical ones must do so.

Next, we formalize the relation between a bundle set and an entire class of argumentation trees. Intuitively, two argumentation trees are equivalent if they are built from the same bundle set (following Definition 3.5); such trees thus constitute a class, and there is a mapping (a one-to-one correspondence) between each bundle set and the class built from it.

Definition 3.6. Let P be a PreDeLP Program and let $Trees(\mathcal{A})$ be the set of all argumentation trees rooted in argument $\mathcal{A}$ in P. Let $\mathcal{T}(\mathcal{A}), \mathcal{T}'(\mathcal{A}) \in Trees(\mathcal{A})$. $\mathcal{T}(\mathcal{A})$ and $\mathcal{T}'(\mathcal{A})$ are *equivalent*, denoted $\mathcal{T}(\mathcal{A}) \equiv \mathcal{T}'(\mathcal{A})$, if both are built from the same bundle set $\mathbb{L}$ for $\mathcal{A}$ following Definition 3.5.

Definition 3.7. Let P be a PreDeLP Program and let $Bundle(\mathcal{A})$ be the set of all bundle sets for argument $\mathcal{A}$ in P. Let $\mathbb{L} \in Bundle(\mathcal{A})$. We define the mapping $\mathbb{T} : Bundle(\mathcal{A}) \rightarrow \overline{Trees(\mathcal{A})}$ as $\mathbb{T}(\mathbb{L}) =_{def} Class(\mathbb{L})$, where $\overline{Trees(\mathcal{A})}$ is the quotient set of $Trees(\mathcal{A})$ by the equivalence relation $\equiv$, and $Class(\mathbb{L})$ denotes the equivalence class such that all $\mathcal{T} \in Class(\mathbb{L})$ is built from $\mathbb{L}$ following Definition 3.5.

Proposition 1. *For any argument $\mathcal{A}$ in a PreDeLP Program P, the mapping $\mathbb{T}$ is a bijection.*

Proposition 2. *Let P be a PreDeLP program, $\mathbb{L}$ be a bundle set for some argument $\mathcal{A}_1$ in P, $\Lambda = [\mathcal{A}_1, \ldots, \mathcal{A}_i, \ldots, \mathcal{A}_n] \in \mathbb{L}$, and $\mathcal{T}(\mathcal{A}_1) \in Class(\mathbb{L})$. $mark(\mathcal{A}_i, \Lambda) = U$ (resp., D) in $\mathbb{L}$ iff $\mathcal{A}_1, \ldots, \mathcal{A}_i$ is the sequence of labels of the path from $root(\mathcal{T}(\mathcal{A}_1))$ to some node N in $\mathcal{T}(\mathcal{A}_1)$ labeled with $\mathcal{A}_i$ such that $mark(N) = U$ (resp., D).*

Proof. Intuitively, the proof shows that both marking procedures consider the same cases and treat them in the same way.

$\Rightarrow$ Let $\Lambda = [\mathcal{A}_1, \ldots, \mathcal{A}_i] \in \mathbb{L}$. From Definition 3.4, we know that $mark(\mathcal{A}_i, \Lambda) = U$ in $\mathbb{L}$. As $\mathcal{T}(\mathcal{A}_1) \in Class(\mathbb{L})$, from Definition 3.7 we know that $\mathcal{T}(\mathcal{A}_1)$ is built from $\mathbb{L}$ according to Definition 3.5. Then, there exists a leaf node N in $\mathcal{T}(\mathcal{A}_1)$ such that Λ is the sequence of labels of the path from $root(\mathcal{T}(\mathcal{A}_1))$ to N. From Definition 2.11, $mark(N) = U$ in $\mathcal{T}(\mathcal{A}_1)$.

Let $\Lambda = [\mathcal{A}_1, \ldots, \mathcal{A}_i, \mathcal{A}_{i+1}, \ldots, \mathcal{A}_n] \in \mathbb{L}$ with $i < n$, and let $E = \{\Lambda' \in \mathbb{L} \mid [\mathcal{A}_1, \ldots, \mathcal{A}_i]$ be a proper segment of $\Lambda'\}$. From Definition 3.4, $mark(\mathcal{A}_i, \Lambda) = U$ iff for all $\Lambda' \in E$, it holds that $mark(\mathcal{A}_{i+1}, \Lambda') = D$. As $\mathcal{T}(\mathcal{A}_1) \in Class(\mathbb{L})$, from Definition 3.7, $\mathcal{T}(\mathcal{A}_1)$ is built from $\mathbb{L}$ according to Definition 3.5; then, for all $\Lambda' \in E, \mathcal{A}_1, \ldots, \mathcal{A}_i, \mathcal{A}_{i+1}, \ldots, \mathcal{A}_n$ is the sequence of labels of a path from $root(\mathcal{T}(\mathcal{A}_1))$ to a leaf node. From Definition 2.11, an inner node N labeled with $\mathcal{A}_i$ is marked U in $\mathcal{T}(\mathcal{A}_1)$ iff all its children are marked as D. From Definition 3.4, $mark(\mathcal{A}_i, \Lambda) = D$ iff there exists $\Lambda' \in E$ such that $mark(\mathcal{A}_{i+1}, \Lambda') = U$. As $\mathcal{T}(\mathcal{A}_1) \in Class(\mathbb{L})$, from Definition 3.7, $\mathcal{T}(\mathcal{A}_1)$ is built from $\mathbb{L}$ according to Definition 3.5; then, for all $\Lambda' \in E, \mathcal{A}_1, \ldots, \mathcal{A}_i, \mathcal{A}_{i+1}, \ldots, \mathcal{A}_n$ is the sequence of labels of a path from $root(\mathcal{T}(\mathcal{A}_1))$ to a leaf node. From Definition 2.11, an inner node N labeled with $\mathcal{A}_i$ is marked D in $\mathcal{T}(\mathcal{A}_1)$ iff (at least) one of its children is marked U.

$\Leftarrow$ An analogous analysis can be done from $\mathcal{T}(\mathcal{A}_1)$ to $\mathbb{L}$, *i.e.*, considering that the node N labeled with $\mathcal{A}_i$ is a leaf (resp., inner node) in $\mathcal{T}(\mathcal{A}_1)$. For reasons of space, we do not include it here. □

Proposition 3. *Let α be a literal and let P be a PreDeLP program. $P \vdash_{\mathcal{W}} \alpha$ iff there exists the exhaustive bundle set $\mathbb{L}$ for some argument $\langle \mathcal{A}, \alpha \rangle$ such that $mark(root(\mathbb{L})) = U$.*

Proof.

$\Rightarrow$ Let $P \vdash_{\mathcal{W}} \alpha$. Then, there exists a dialectical tree $\mathcal{T}(\mathcal{A})$ for some argument $\langle \mathcal{A}, \alpha \rangle$ such that $mark(root(\mathcal{T}(\mathcal{A}))) = U$. From Proposition 1 we know that there exists a bundle set $\mathbb{L}$ such that $\mathcal{T}(\mathcal{A}) \in Class(\mathbb{L})$; *i.e.*, $\mathcal{T}(\mathcal{A})$ is built from $\mathbb{L}$ according to Definition 3.5. Furthermore, $\mathbb{L}$ is exhaustive (because $\mathcal{T}(\mathcal{A})$ is a dialectical tree). From Proposition 2, $mark(root(\mathbb{L})) = U$.

$\Leftarrow$ Let $\mathbb{L}$ be the exhaustive bundle set for some argument $\langle \mathcal{A}, \alpha \rangle$ such that $mark(root(\mathbb{L})) = U$. Let $\mathcal{T}(\mathcal{A}) \in Class(\mathbb{L})$ be an argumentation tree built from $\mathbb{L}$ according to Definition 3.5. As $\mathbb{L}$ is exhaustive, $\mathcal{T}(\mathcal{A})$ is then a dialectical tree. From Proposition 2, $mark(root(\mathcal{T}(\mathcal{A}))) = U$. Then, from Definition 2.12, $P \vdash_{\mathcal{W}} \alpha$. □

3.2 Minimality of Bundle Sets

Next, we introduce the necessary concepts to define a particular type of bundle set, called *parsimonious*, that captures our notion of minimality, understood as the minimal information (according to set inclusion) necessary to warrant a literal.

In our approach, we can relate bundle sets through a specially defined inclusion relation. Intuitively, given two bundle sets $\mathbb{L}$ and $\mathbb{L}'$, $\mathbb{L}'$ *includes* $\mathbb{L}$ if every line in $\mathbb{L}$ is a segment of some line in $\mathbb{L}'$. Next, based on Definition 3.1, we are going to formalize this relation, and the following two definitions introduce the *expansion* and *contraction* operators of a bundle set by an argumentation line.

Definition 3.8. Let $\mathbb{L}$ and $\mathbb{L}'$ be two bundle sets rooted in argument $\mathcal{A}$ from a PreDeLP program P. We define the *bundle set inclusion* relation, denoted $\sqsubseteq$, as a subset of $Bundle(\mathcal{A}) \times Bundle(\mathcal{A})$. We denote with $\mathbb{L} \sqsubseteq \mathbb{L}'$ that for all $\Lambda \in \mathbb{L}$ there exist some $\Lambda' \in \mathbb{L}'$ such that Λ is a segment of Λ'. If furthermore, there exists at least some $\Lambda \in \mathbb{L}$ that is a proper segment for some $\Lambda' \in \mathbb{L}'$, then the inclusion relation is strict, denote as $\mathbb{L} \sqsubset \mathbb{L}'$.

Definition 3.9. Let $\mathbb{L}$ and Λ' be a bundle set and a line, respectively, both rooted in argument $\mathcal{A}$ from a PreDeLP program P. The *expansion* of $\mathbb{L}$ by Λ', denoted as $\mathbb{L} \oplus \Lambda'$, is defined as follows:

1. If there exists $\Lambda \in \mathbb{L}$ such that it is a proper segment of Λ', then $\mathbb{L} \oplus \Lambda' = (\mathbb{L} \setminus \Lambda) \cup \Lambda'$.

[1] The functions *head*() and *tail*() have the usual meaning as in list processing, returning the first element in a list and the list formed by all elements except the first, resp.

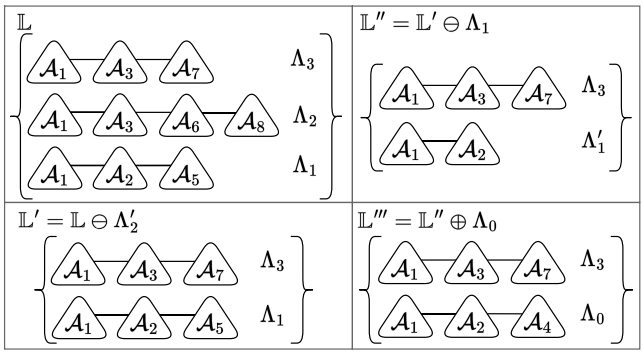

Figure 2: Illustrating contraction and expansion on bundle sets (Example 3).

2. If there is no $\Lambda \in \mathbb{L}$ that is a proper segment of Λ', and Λ' is not a segment of any $\Lambda \in \mathbb{L}$, then $\mathbb{L} \oplus \Lambda' = \mathbb{L} \cup \Lambda'$.
3. Otherwise, $\mathbb{L} \oplus \Lambda' = \mathbb{L}$.

The expansion of $\mathbb{L}$ by Λ' produces a new bundle set $\mathbb{L} \oplus \Lambda'$ such that $\mathbb{L} \sqsubseteq \mathbb{L} \oplus \Lambda'$ and Λ' is a segment of some $\Lambda \in \mathbb{L} \oplus \Lambda'$.

Intuitively, condition 1 says that if Λ' has as proper segment Λ in $\mathbb{L}$, then the expansion removes Λ and introduces Λ' into $\mathbb{L}$. Condition 2 states that if Λ' neither has as proper segment Λ in $\mathbb{L}$ nor it is a segment of any line in $\mathbb{L}$, then Λ' is a new line that is directly added to the bundle set. Condition 3 captures those cases where Λ' is a segment of a line in $\mathbb{L}$ and then the original bundle set remains unchanged because Λ' is already contained in $\mathbb{L}$.

Definition 3.10. Let $\mathbb{L}$ and Λ' be a bundle set and a line, respectively, both rooted in argument $\mathcal{A}$ from a PreDeLP program P. The *contraction* of $\mathbb{L}$ by Λ', denoted as $\mathbb{L} \ominus \Lambda'$, is defined as follows:

1. If $\Lambda' \in \mathbb{L}$, then $\mathbb{L} \ominus \Lambda' = (\mathbb{L} \setminus \Lambda') \cup \Lambda''$, if there exists Λ'' which is the longest argumentation line that is a proper segment of Λ' and is not a proper segment of any other $\Lambda \in (\mathbb{L} \setminus \Lambda')$. If such Λ'' does not exists then $\mathbb{L} \ominus \Lambda' = (\mathbb{L} \setminus \Lambda')$.
2. If Λ' is a proper segment of some $\Lambda \in \mathbb{L}$, then $\mathbb{L} \ominus \Lambda' = (\mathbb{L} \setminus \mathbb{S}) \cup \Lambda''$, where $\mathbb{S} = \{\Lambda \mid \Lambda \in \mathbb{L}$ and Λ' is a proper segment of $\Lambda\}$ and Λ'' is (if there exists) the longest line that is a proper segment of Λ' and is not a proper segment of any other $\Lambda \in (\mathbb{L} \setminus \mathbb{S})$. If such Λ'' does not exists then $\mathbb{L} \ominus \Lambda' = (\mathbb{L} \setminus \mathbb{S})$.
3. Otherwise, $\mathbb{L} \ominus \Lambda' = \mathbb{L}$.

The contraction of $\mathbb{L}$ by Λ' produces a new bundle set $\mathbb{L} \ominus \Lambda'$ such that $\mathbb{L} \ominus \Lambda' \sqsubseteq \mathbb{L}$ and $\Lambda' \notin \mathbb{L} \ominus \Lambda'$.

Condition 1 says that if Λ' is in $\mathbb{L}$, then the contraction removes Λ'. Condition 2 states that for all argumentation lines Λ in $\mathbb{L}$ such that Λ' is as proper segment of Λ, the contraction removes Λ from $\mathbb{L}$. Conditions 1 and 2 add to $\mathbb{L}$ the longest argumentation line Λ'' that is a proper segment of Λ' because contraction makes the minimal change in the original bundle set to satisfy the condition $\Lambda' \notin \mathbb{L} \ominus \Lambda'$. However, Λ'' is added just in those cases where Λ'' is not a proper segment of some line in $\mathbb{L}$, otherwise, the result of the contraction will be not a bundle set. Condition 3 captures those cases where Λ' is not a segment of an argumentation line in $\mathbb{L}$, then the line cannot be removed from the bundle set.

The expansion and contraction operations allow us to declaratively describe changes in the bundle sets, while inclusion allows us to establish a relationship between the different bundle sets that arise from distinct expansion and contraction operations. Note that expansion and contraction on bundle sets are operations in which we operate not only with elements of the set (the argumentation lines that belong to the bundle set) but also with "parts" of those elements (the segments of the argumentation lines in the bundle set). Also, if we think of the argumentation lines in a bundle set as different lines of dispute, where the main topic of the debate is the root argument, the expansion operation allows us to represent all the possible ways in which the debate can evolve.

Example 3. *Let's recall the arguments and relations presented in Example 2; Figure 2 illustrates this example. All possible exhaustive lines rooted in* $\mathcal{A}_1$ *are:* $\Lambda_0 = [\mathcal{A}_1, \mathcal{A}_2, \mathcal{A}_4]$, $\Lambda_1 = [\mathcal{A}_1, \mathcal{A}_2, \mathcal{A}_5]$, $\Lambda_2 = [\mathcal{A}_1, \mathcal{A}_3, \mathcal{A}_6, \mathcal{A}_8]$ *and* $\Lambda_3 = [\mathcal{A}_1, \mathcal{A}_3, \mathcal{A}_7]$.

Let $\mathbb{L} = \{\Lambda_1, \Lambda_2, \Lambda_3\}$ *and assume that we want to contract* $\mathbb{L}$ *by* $\Lambda'_2 = [\mathcal{A}_1, \mathcal{A}_3, \mathcal{A}_6]$. *As* Λ'_2 *is a proper segment of* Λ_2, *then* $\mathbb{L} \ominus \Lambda'_2 = (\mathbb{L} \setminus \{\Lambda_2\}) \cup \Lambda''_2$ *where* $\{\Lambda_2\}$ *is the set of all lines in the bundle set that has* Λ'_2 *as a proper segment, and where* Λ''_2 *does not exist because the longest proper segment of* Λ'_2, *that is* $[\mathcal{A}_1, \mathcal{A}_3]$, *is also a proper segment of* Λ_3. *Then, we have that* $\mathbb{L}' = \mathbb{L} \ominus \Lambda'_2 = \{\Lambda_1, \Lambda_3\}$.

Assume now that we want to contract $\mathbb{L}'$ *by* Λ_1. *In this case, as* $\Lambda_1 \in \mathbb{L}'$ *then* $\mathbb{L}' \ominus \Lambda_1 = (\mathbb{L}' \setminus \Lambda_1) \cup \Lambda'_1$ *where* $\Lambda'_1 = [\mathcal{A}_1, \mathcal{A}_2]$ *is the longest proper segment of* Λ_1 *such that* Λ'_1 *is not a proper segment of any other line in* $\mathbb{L}' \setminus \Lambda_1$. *Then,* $\mathbb{L}'' = \mathbb{L}' \ominus \Lambda_1 = \{\Lambda'_1, \Lambda_3\}$.

Finally, assume that we want to expand $\mathbb{L}''$ *by* Λ_0. *As* Λ_0 *has as proper segment some line in* $\mathbb{L}''$, *that is,* Λ'_1 *is a proper segment of* Λ_0, *then* $\mathbb{L}'' \oplus \Lambda_0 = (\mathbb{L}'' \setminus \Lambda'_1) \cup \Lambda_0$. *Then,* $\mathbb{L}''' = \mathbb{L}'' \oplus \Lambda_0 = \{\Lambda_0, \Lambda_3\}$.

The concepts of expansion and contraction will also help us to define the notion that we need to be able to capture the idea of a parsimonious bundle set. Intuitively, the requirement for a bundle set to be parsimonious is that it yields the same information as the exhaustive bundle set about the warrant status of the root argument while being minimal. We express this idea formally through the notion of *relevance*. This concept was first introduced by Prakken et. al. in [18] and further studied in [19] in the context of argument games and dialogue games, respectively. According to [19], a move is relevant in a dialogue iff it changes the status of the dialogue's initial move. Mapping our context into theirs, arguments correspond to moves, argumentation lines represent lines of discussion/dispute, and bundle sets correspond to (argumentative) dialogues. In Prakken's approach, relevance is a property subject to the relationship between an argument and a dialogue. In our approach, relevance is a notion associated with (each argument in) an exhaustive argumentation line and its relation with the rest of the lines in a bundle set. We consider (each argument in) an exhaustive argumentation line because in this way we can take into account all moves that belong to that dispute (if the line were partial there could be moves that we are not considering and that could affect our analysis), and we consider the relationship of each argumentation line with the other lines in the bundle set since the warrant status of the root argument depends on that.

Intuitively, an argumentation line is relevant to a bundle set if it constitutes a line of dispute, understanding dispute as an exchange of arguments between a proponent and an opponent, such that the information discussed in it contributes to the general debate, *i.e.*, the

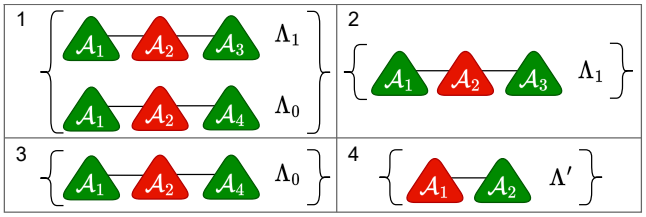

Figure 3: Illustrating relevance (Example 4).

dispute changes the warrant status of the claim under discussion. Formally, we have:

Definition 3.11. Let $\mathbb{L}$ and Λ be a bundle set and an exhaustive argumentation line, respectively, both rooted in argument $\mathcal{A}$ from a PreDeLP program P. We say that Λ is *relevant* for $\mathbb{L}$ if:

- $\Lambda \notin \mathbb{L}$, and there exists a segment Λ' of Λ such that $mark(root(\mathbb{L})) \neq mark(root(\mathbb{L} \oplus \Lambda'))$, or
- $\Lambda \in \mathbb{L}$, and $mark(root(\mathbb{L})) \neq mark(root(\mathbb{L} \ominus \Lambda))$.

Otherwise, Λ is *irrelevant* for $\mathbb{L}$.

An argumentation line is relevant (resp., irrelevant) for a bundle set according to its contribution to the warrant status of the root argument. Then, the notion of relevance is tied to a specific bundle set, *i.e.*, an argumentation line may be relevant for a given bundle set and irrelevant for another. The next example illustrates two interesting cases of relevance.

Example 4. *The following example is illustrated in Figure 3. Consider a PreDeLP program P where the following set of arguments can be built: $\{\mathcal{A}_1, \mathcal{A}_2, \mathcal{A}_3, \mathcal{A}_4\}$, and where (only) the following defeat relations hold: $(\mathcal{A}_2, \mathcal{A}_1)$, $(\mathcal{A}_3, \mathcal{A}_2)$, $(\mathcal{A}_4, \mathcal{A}_2)$. All possible exhaustive lines rooted in $\mathcal{A}_1$ are: $\Lambda_0 = [\mathcal{A}_1, \mathcal{A}_2, \mathcal{A}_4]$, and $\Lambda_1 = [\mathcal{A}_1, \mathcal{A}_2, \mathcal{A}_3]$.*

Let $\mathbb{L} = \{\Lambda_0, \Lambda_1\}$ be the exhaustive bundle set shown in part (1). Lines Λ_0 and Λ_1 are irrelevant to $\mathbb{L}$ since for each there exists a different $\Lambda_i \in \mathbb{L}$ that contributes to maintaining the warrant status of $\mathcal{A}_1$. If we contract $\mathbb{L}$ by Λ_0, shown in part (2), or by Λ_1, shown in part (3), the warrant status of $\mathcal{A}_1$ is maintained. However, if after contracting by Λ_1 (3), we contract the resulting bundle set $\mathbb{L}'$ by Λ_0 (4) the warrant status of $\mathcal{A}_1$ changes because Λ_0 is relevant for $\mathbb{L}'$. A similar situation occurs if, after contracting by Λ_0 (2), we contract the resulting bundle set $\mathbb{L}'$ by Λ_1 (4) the warrant status of $\mathcal{A}_1$ changes because Λ_1 is relevant for $\mathbb{L}'$.

Let $\mathbb{L}''$ be the bundle set shown in part (4). Both Λ_0 and Λ_1 are relevant for $\mathbb{L}''$. However, if we expand it by Λ_1 (resp., Λ_0), for the resulting bundle set $\mathbb{L}'''$ (2) (resp., 3) the argumentation line Λ_0 (resp., Λ_1) is irrelevant.

Definition 3.12. Let $\mathbb{L}$ be a bundle set for argument $\mathcal{A}$ from a PreDeLP program P. $\mathbb{L}$ is *parsimonious* if all, and no other, relevant argumentation lines for $\mathbb{L}$ belong to $\mathbb{L}$. Otherwise, $\mathbb{L}$ is *non-parsimonious*.

The set of all bundle sets rooted in an argument represents all possible states in the debate about that argument. Intuitively, a parsimonious bundle set represents a state of the debate in which some lines of discussion have been explored to their final consequences, understanding this as no new arguments can be provided in these lines, where none of these lines is irrelevant, and where a conclusion has been reached about the topic of debate. Then, additional arguments will not change the status of the claim under discussion.

The bundle set $\mathbb{L}' = \{\Lambda_1, \Lambda_3\}$ shown in Figure 1 is parsimonious because it contains the minimum information necessary to unambiguously determine the warrant status of the root argument; that is, if $\mathbb{L}'$ is expanded by another (exhaustive) line, the root state will not change, and if $\mathbb{L}'$ is contracted by Λ_1 or Λ_3, the status of the root argument will change.

Note that there may exist several parsimonious bundle sets for the same argument. The bundle sets shown in parts (2) and (3) in Figure 3 are both parsimonious bundle sets for the same argument $\mathcal{A}_1$. Also note that a bundle set may be exhaustive and parsimonious, partial and parsimonious, or partial and non-parsimonious.

The following proposition formalizes that parsimonious bundle sets are minimal bundle sets that allow to unequivocally identify the warrant status of the root argument.

Proposition 4. *Let $\mathbb{L}$ be a bundle set for argument $\mathcal{A}$ from a PreDeLP Program P. Then, $\mathbb{L}$ is parsimonious iff $\mathbb{L}$ is a minimal (w.r.t bundle set inclusion $\sqsubseteq$) bundle set such that for any bundle set $\mathbb{L}'$ for $\mathcal{A}$ such that $\mathbb{L} \sqsubseteq \mathbb{L}'$, it holds that $mark(root(\mathbb{L})) = mark(root(\mathbb{L}'))$.*

Proof.

$\Rightarrow$ Let $\mathbb{L}$ be a parsimonious bundle set for $\mathcal{A}$. Then, $\mathbb{L}$ contains all and only those argumentation lines Λ relevant to it. Then, it must hold that: (1) for all $\Lambda \notin \mathbb{L}$, there does not exist a segment Λ' of Λ such that $mark(root(\mathbb{L})) \neq mark(root(\mathbb{L} \oplus \Lambda'))$; and (2) for all $\Lambda \in \mathbb{L}$, $mark(root(\mathbb{L})) \neq mark(root(\mathbb{L} \ominus \Lambda))$. For (1), it holds that $mark(root(\mathbb{L})) = mark(root(\mathbb{L}'))$ for all $\mathbb{L}'$ such that $\mathbb{L} \sqsubseteq \mathbb{L}'$, and for (2) $\mathbb{L}$ is the minimal bundle set (w.r.t. $\sqsubseteq$) satisfying the previous property.

$\Leftarrow$ Let $\mathbb{L}$ be a minimal (w.r.t. $\sqsubseteq$) bundle set for $\mathcal{A}$ such that for any bundle set $\mathbb{L}'$ such that $\mathbb{L} \sqsubseteq \mathbb{L}'$, it holds that $mark(root(\mathbb{L})) = mark(root(\mathbb{L}'))$. Then, (1) for all $\Lambda \notin \mathbb{L}$, there does not exist a segment Λ' of Λ such that $mark(root(\mathbb{L})) \neq mark(root(\mathbb{L} \oplus \Lambda'))$; and (2) for all $\Lambda \in \mathbb{L}$, $mark(root(\mathbb{L})) \neq mark(root(\mathbb{L} \ominus \Lambda))$. For (1) and (2), $\mathbb{L}$ contains all and only those lines relevant to it. Then, from Definition 3.12, $\mathbb{L}$ is parsimonious. □

Using Proposition 4, it is straightforward to show that exhaustive and parsimonious bundle sets are equivalent concerning their root arguments' warrant status.

Proposition 5. *Let $\mathbb{L}$ and $\mathbb{L}'$ be the exhaustive and a parsimonious bundle set, respectively, both for some argument $\mathcal{A}$ in a PreDeLP program P. Then, $mark(root(\mathbb{L})) = mark(root(\mathbb{L}'))$.*

Proof. Since $\mathbb{L}$ is the exhaustive bundle set for argument $\mathcal{A}$, then $\mathbb{L}' \sqsubseteq \mathbb{L}$. As $\mathbb{L}'$ is a parsimonious bundle set for $\mathcal{A}$, from Proposition 4 we know that for all bundle set $\mathbb{L}''$ such that $\mathbb{L}' \sqsubseteq \mathbb{L}''$ it holds that $mark(root(\mathbb{L}')) = mark(root(\mathbb{L}''))$. As consequence, $mark(root(\mathbb{L}')) = mark(root(\mathbb{L}))$. □

Finally, the previous result allows us to define the warrant procedure through parsimonious bundle sets.

Proposition 6. *Let α be a literal and let P be a PreDeLP program. $P \vdash_W \alpha$ iff there exists a parsimonious bundle set $\mathbb{L}$ for some argument $\langle \mathcal{A}, \alpha \rangle$ such that $mark(root(\mathbb{L})) = U$.*

Proof.

$\Rightarrow$ Let's assume that $P \mid\sim_{\mathcal{W}} \alpha$. Then, we know from Proposition 3 that there exists the exhaustive bundle set $\mathbb{L}'$ for some $\langle \mathcal{A}, \alpha \rangle$ such that $mark(root(\mathbb{L}')) = U$. As $\mathbb{L}'$ is exhaustive, then there exists a parsimonious bundle set $\mathbb{L}$ for $\langle \mathcal{A}, \alpha \rangle$ such that $\mathbb{L} \sqsubseteq \mathbb{L}'$. Then, by Proposition 5, $mark(root(\mathbb{L})) = mark(root(\mathbb{L}')) = U$.
$\Leftarrow$ Let's assume that $\mathbb{L}$ is a parsimonious bundle set for some $\langle \mathcal{A}, \alpha \rangle$, and let $mark(root(\mathbb{L})) = U$. Let $\mathbb{L}'$ be the exhaustive bundle set for argument $\langle \mathcal{A}, \alpha \rangle$. We know from Proposition 5 that $mark(root(\mathbb{L}')) = mark(root(\mathbb{L})) = U$. Then, from Proposition 3 it follows that $P \mid\sim_{\mathcal{W}} \alpha$. □

4 CONCLUSIONS AND FUTURE WORK

In this work, we addressed the problem of declaratively characterizing the concept of warrant in PreDeLP. Toward this end, we defined a novel structure, called parsimonious bundle set, that is more general, yet minimal, than dialectical trees. This notion captures the minimal information needed to warrant a claim. The main motivation behind this work was to study the properties of these structures toward the definition of model-theoretic semantics for PreDeLP, a contribution that we consider to be of value in structured argumentation. From the study of parsimonious bundle sets and the expansion and contraction operations, it follows that, unlike set-theoretic operations in this context, we must consider—in addition to each element of the set—the internal structure of each one of them. In other words, expansion and contraction of bundle sets are operations in which we operate not only with elements of the set (the lines in the bundle set) but also with "parts" of those elements (the segments of the lines in the bundle set).

The development of model-theoretic semantics for PreDeLP is one of our main goals for future work. However, several interesting lines of research are still open, such as the study of the relation between operations defined on bundle sets with Belief Revision operators over argumentative frameworks [2]. Expansion and contraction, apart from having the same names, satisfy some properties similar to the postulates of success and inclusion of the expansion and contraction operations in Belief Revision. Defining an operation similar to that of revision, and describing the entire construction process of a bundle set through Belief Revision-inspired operators, are problems that remain open for structured argumentation frameworks [13, 15, 20] and in particular for PreDeLP. The more practical aspects of the present work, that is, the design of algorithms to compute the different concepts presented so far and their corresponding complexity analysis, is another line of research that remains open.

Acknowledgments

We thank the anonymous reviewers for their comments, which helped improve the final version of this paper. This work was funded in Argentina in part by the following institutions: Universidad Nacional del Sur (UNS) under grants PGI 24/N057 and PGI 24/ZN055, Universidad Nacional de Entre Rios (PDTS-UNER 7066), CONICET under grants PIP 11220210100577CO and PIP 11220200101408CO, and Agencia Nacional de Promocion Cientifica y Tecnologica under grants PICT-2018-0475 (PRH-2014-0007) and PICT-2020-SERIEA-01481. The authors also acknowledge support by the Spanish project PID 2022-139835NBC21 funded by "European Union NextGenerationEU/PRTR", and by MCIN/AEI/10.13039/501100011033.

REFERENCES

[1] Pietro Baroni, Martin Caminada, and Massimiliano Giacomin. 2011. An introduction to argumentation semantics. *Knowl. Eng. Rev.* 26, 4 (2011), 365–410.
[2] Pietro Baroni, Eduardo Fermé, Massimiliano Giacomin, and Guillermo Ricardo Simari. 2022. Belief Revision and Computational Argumentation: A Critical Comparison. *J. Log. Lang. Inf.* 31, 4 (2022), 555–589.
[3] Martin Caminada and Leila Amgoud. 2007. On the evaluation of argumentation formalisms. *Artif. Intell.* 171, 5-6 (2007), 286–310.
[4] Carlos Iván Chesñevar, Ramón F Brena, and José-Luis Aguirre. 2005. Knowledge Distribution in Large Organizations Using Defeasible Logic Programming.. In *Canadian Conference on AI*. Springer, 244–256.
[5] Carlos Iván Chesñevar and Guillermo Ricardo Simari. 2007. A Lattice-Based Approach to Computing Warranted Beliefs in Skeptical Argumentation Frameworks. In *IJCAI 2007, Proceedings of the 20th International Joint Conference on Artificial Intelligence, Hyderabad, India, January 6-12, 2007*, Manuela M. Veloso (Ed.). 280–285.
[6] Carlos Iván Chesñevar, Guillermo Ricardo Simari, and Lluís Godo. 2005. Computing Dialectical Trees Efficiently in Possibilistic Defeasible Logic Programming. In *Logic Programming and Nonmonotonic Reasoning, 8th International Conference, LPNMR 2005, Diamante, Italy, September 5-8, 2005, Proceedings (Lecture Notes in Computer Science, Vol. 3662)*, Chitta Baral, Gianluigi Greco, Nicola Leone, and Giorgio Terracina (Eds.). Springer, 158–171.
[7] Boris Galitsky and Eugene William McKenna. 2017. Sentiment extraction from consumer reviews for providing product recommendations. US Patent 9,646,078.
[8] Alejandro Javier García and Guillermo Ricardo Simari. 2004. Defeasible Logic Programming: An Argumentative Approach. *Theory Pract. Log. Program.* 4, 1-2 (2004), 95–138.
[9] Alejandro Javier García and Guillermo Ricardo Simari. 2014. Defeasible logic programming: DeLP-servers, contextual queries, and explanations for answers. *Argument Comput.* 5, 1 (2014), 63–88.
[10] Pascal Hitzler and Anthony Karel Seda. 2011. *Mathematical Aspects of Logic Programming Semantics*. CRC Press.
[11] Robert Koons. 2022. Defeasible Reasoning. In *The Stanford Encyclopedia of Philosophy* (Summer 2022 ed.), Edward N. Zalta (Ed.). Metaphysics Research Lab, Stanford University.
[12] Mario A. Leiva, Alejandro Javier García, Paulo Shakarian, and Gerardo I. Simari. 2022. Argumentation-Based Query Answering under Uncertainty with Application to Cybersecurity. *Big Data Cogn. Comput.* 6, 3 (2022), 91.
[13] Alejandro J. García Marcelo A. Falappa and Guillermo R. Simari. 2023. Merging operators on stratified belief bases equipped with argumentative inference. *Journal of Applied Non-Classical Logics* 33, 3-4 (2023), 387–420.
[14] Maria Vanina Martinez, Alejandro Javier García, and Guillermo Ricardo Simari. 2012. On the Use of Presumptions in Structured Defeasible Reasoning. In *Computational Models of Argument - Proceedings of COMMA 2012, Vienna, Austria, September 10-12, 2012 (Frontiers in Artificial Intelligence and Applications, Vol. 245)*, Bart Verheij, Stefan Szeider, and Stefan Woltran (Eds.). IOS Press, 185–196.
[15] Martín O. Moguillansky, Nicolás D. Rotstein, Marcelo A. Falappa, Alejandro Javier García, and Guillermo Ricardo Simari. 2013. Dynamics of knowledge in *DeLP* through Argument Theory Change. *Theory Pract. Log. Program.* 13, 6 (2013), 893–957.
[16] Eric Nunes, Paulo Shakarian, and Gerardo I. Simari. 2018. At-risk system identification via analysis of discussions on the darkweb. In *2018 APWG Symposium on Electronic Crime Research, eCrime 2018, San Diego, CA, USA, May 15-17, 2018*. IEEE, 1–12.
[17] John L Pollock. 1987. Defeasible reasoning. *Cogn. Sci.* 11, 4 (1987), 481–518.
[18] Henry Prakken. 2001. Relating protocols for dynamic dispute with logics for defeasible argumentation. *Synthese* 127, 1 (2001), 187–219.
[19] Henry Prakken. 2005. Coherence and flexibility in dialogue games for argumentation. *Journal of logic and computation* 15, 6 (2005), 1009–1040.
[20] Henry Prakken. 2023. Relating Abstract and Structured Accounts of Argumentation Dynamics: the Case of Expansions. In *Proceedings of the 20th International Conference on Principles of Knowledge Representation and Reasoning, KR 2023, Rhodes, Greece, September 2-8, 2023*, Pierre Marquis, Tran Cao Son, and Gabriele Kern-Isberner (Eds.). 562–571.
[21] Guillermo R Simari and Ronald P Loui. 1992. A mathematical treatment of defeasible reasoning and its implementation. *Artif. Intell.* 53, 2-3 (1992), 125–157.
[22] Frieder Stolzenburg, Alejandro J García, Carlos I Chesñevar, and Guillermo R Simari. 2003. Computing generalized specificity. *J. Appl. Non-Class. Log.* 13, 1 (2003), 87–113.