On Learning Abstract Argumentation Graphs from Bivalent Statement Labellings

Régis Riveret
Data61 - CSIRO
Spring Hill, Australia
Email: regis.riveret@data61.csiro.au

***Abstract*—In this paper, we investigate the problem of finding argumentation graphs consistent with some observed statement labellings. We consider a general abstract framework, where the structure of arguments is left unspecified, and we focus on particular grounded argument labellings where arguments can be omitted. The specification of such grounded labellings, the Principle of Multiple Explanations and the Principle of Parsimony lead us to a simple and efficient anytime algorithm to induce 'on the fly' all the 'argument-parsimonious' argumentation graphs consistent with an input stream of statement labellings.**

I. Introduction

Computational models of argumentation are the focus of many research programmes in artificial intelligence, with application domains ranging from law to autonomous agents and multi-agent systems [4], [9].

Argumentation graphs (or frameworks) [5] are the prevalent abstract basis of many computational models of argumentation. Given an argumentation graph, arguments can be labelled according to the specification of some argument acceptance labellings, and from these argument labellings, statements can be labelled according to the specification of some statement acceptance labellings [2]. In this paper, we investigate the inverse problem, namely, the problem of learning the structure of argumentation graphs consistent with an observed set of statement labellings.

To guide us, we will consider Epicurus' Principle of Multiple Explanations (according to which if several explanations are consistent with the observed data then we should retrain them all) and Ockham's Principle of Parsimony (according to which "Entities should not be multiplied beyond necessity"). Consequently and more precisely, we will tackle the problem of learning the structure of all the argument-parsimonious argumentation graphs consistent with an observed set of statement labellings.

Our argumentation learning problem is related to some recent problems in argumentation, such as the 'enforcement' problem [3], i.e. the modification of an argumentation graph so that some arguments obtain particular status in the modified argumentation graph (possibly with a minimum amount of modifications), or the 'realisability' problem [6], regarding whether a set of sets of arguments can be 'realised' by an argumentation graph along with a semantics, or the 'synthesis' problem [8] which regards the synthesis of argumentation graphs that are semantically closest to the given knowledge, or the 'learning structure' problem addressed in [10] which regards the determination of an argumentation graph to account for samples of argument labellings in a probabilistic setting.

In contrast to the above-mentioned problems, we observe statement labellings, and instead of inducing one particular argumentation graph consistent with argument labellings, we look for an anytime algorithm inducing 'on the fly' all argument-parsimonious argumentation graphs consistent with the statement labellings.

To address our problem, we will inverse the reasoning typically used in argumentation multi-labelling systems [2]: from the observation of statement acceptance labellings, we consider argument acceptance labellings, from which we induce argumentation graphs (meant to account for the observed statement labellings).

This paper is organised as follows. Section II introduces the abstract argumentation setting. Section III specifies the problem we address. Then, we investigate the induction of argumentation graphs in Section IV, before concluding.

II. Argumentation Setting

To account for arguments and attacks between arguments in an abstract way, the argumentation setting is built on abstract argumentation graphs [5].

Definition 1 (Argumentation graph)**.** *An* argumentation graph *is a pair $\langle \mathcal{A}, \rightsquigarrow \rangle$ where $\mathcal{A}$ is a set of arguments, and $\rightsquigarrow \subseteq \mathcal{A} \times \mathcal{A}$ is a binary relation of attack.*

Notation 1. *Given an argumentation graph $\mathcal{G} = \langle \mathcal{A}, \rightsquigarrow \rangle$, we may denote $\mathcal{A}$ and $\rightsquigarrow$ as $\mathcal{A}_\mathcal{G}$ and $\rightsquigarrow_\mathcal{G}$, respectively.*

Definition 2 (Subgraph)**.** *A* subgraph $\mathcal{H}$ *of an argumentation graph $\mathcal{G}$ is an argumentation graph such that $\mathcal{A}_\mathcal{H} \subseteq \mathcal{A}_\mathcal{G}$, and $\forall A, B \in \mathcal{A}_\mathcal{H}$, $(A \rightsquigarrow_\mathcal{G} B)$ iff $(A \rightsquigarrow_\mathcal{H} B)$.*

Thus $\mathcal{H}$ is a subgraph of $\mathcal{G}$ if it has exactly the attacks that appear in $\mathcal{G}$ over the same set of arguments.

Given an argumentation graph, the sets of arguments that are accepted or not, that is, those arguments that shall survive or not to possible attacks, are computed using some semantics, and, for our purposes, we will consider Dung's grounded semantics [5] by labelling arguments as in [12] and [11] (a slight adaptation of the labellings reviewed in [1] for a probabilistic setting). So, we will distinguish $\{\mathsf{IN}, \mathsf{OUT}, \mathsf{UND}\}$-labellings and $\{\mathsf{IN}, \mathsf{OUT}, \mathsf{UND}, \mathsf{OFF}\}$-labellings. In a $\{\mathsf{IN}, \mathsf{OUT}, \mathsf{UND}\}$-labelling,

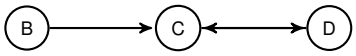


Fig. 1. An argumentation graph. Argument B attacks argument C, arguments C and D attack each other.

Fig. 2. The graph on the left is a subgraph of the graph in Figure 1, while the graph on the right is not one of its subgraphs.

each argument is associated with one label which is either IN, OUT, UND, respectively meaning that the argument is justified, rejected, or undecided. In a $\{\text{IN}, \text{OUT}, \text{UND}, \text{OFF}\}$-labelling, the label 'OFF' indicates that the argument is omitted, for example when it is not expressed in a dialogue.

Definition 3 (Argument labelling)**.** *Let $\mathcal{G}$ denote an argumentation graph.*

- *A* $\{\text{IN}, \text{OUT}, \text{UND}\}$-labelling *of $\mathcal{G}$ is a total function* $\text{L} : \mathcal{A}_\mathcal{G} \rightarrow \{\text{IN}, \text{OUT}, \text{UND}\}$.
- *A* $\{\text{IN}, \text{OUT}, \text{UND}, \text{OFF}\}$-labelling *of $\mathcal{G}$ is a total function* $\text{L} : \mathcal{A}_\mathcal{G} \rightarrow \{\text{IN}, \text{OUT}, \text{UND}, \text{OFF}\}$.

Notation 2.

- *We write* $\text{IN}(\text{L})$ *for* $\{A|\text{L}(A) = \text{IN}\}$, $\text{OUT}(\text{L})$ *for* $\{A|\text{L}(A) = \text{OUT}\}$, $\text{UND}(\text{L})$ *for* $\{A|\text{L}(A) = \text{UND}\}$, *and* $\text{OFF}(\text{L})$ *for* $\{A|\text{L}(A) = \text{OFF}\}$.
- *A* $\{\text{IN}, \text{OUT}, \text{UND}\}$*-labelling* L *is represented as a tuple* $\langle \text{IN}(\text{L}), \text{OUT}(\text{L}), \text{UND}(\text{L})\rangle$, *and a* $\{\text{IN}, \text{OUT}, \text{UND}, \text{OFF}\}$*-labelling* L *as a tuple* $\langle \text{IN}(\text{L}), \text{OUT}(\text{L}), \text{UND}(\text{L}), \text{OFF}(\text{L})\rangle$.

Most of the interesting argument labellings are assumed to be complete. An argumentation graph may have several complete $\{\text{IN}, \text{OUT}, \text{UND}\}$-labellings: we will focus on the unique complete labelling with the smallest set of labels IN, i.e. the grounded $\{\text{IN}, \text{OUT}, \text{UND}\}$-labelling, see [1].

Definition 4 (Complete $\{\text{IN}, \text{OUT}, \text{UND}\}$-labelling)**.** *Let $\mathcal{G}$ denote an argumentation graph. A* complete $\{\text{IN}, \text{OUT}, \text{UND}\}$-labelling *of $\mathcal{G}$ is a* $\{\text{IN}, \text{OUT}, \text{UND}\}$*-labelling such that for every argument A in $\mathcal{A}_\mathcal{G}$ it holds that:*

1) A is labelled IN *iff all attackers of A are labelled* OUT,

2) A is labelled OUT *iff A has an attacker that is labelled* IN.

Definition 5 (Grounded $\{\text{IN}, \text{OUT}, \text{UND}\}$-labelling)**.** *A complete labelling* L *is a* grounded $\{\text{IN}, \text{OUT}, \text{UND}\}$-labelling *of an argumentation graph $\mathcal{G}$ if* $\text{IN}(\text{L})$ *is minimal (w.r.t. set inclusion) amongst all complete* $\{\text{IN}, \text{OUT}, \text{UND}\}$*-labellings of $\mathcal{G}$.*

Since a complete labelling is a total function, if an argument is not labelled IN or OUT, then it is labelled UND.

The reason to focus on the grounded $\{\text{IN}, \text{OUT}, \text{UND}\}$-labelling is that it is – well – grounded, and because given an argumentation graph, it is possible to compute it in a polynomial time, as in Algorithm 1 [7].

Algorithm 1 begins by labelling IN all the arguments not being attacked or whose attackers are OUT (line 4), and then it iteratively labels OUT any argument attacked by an argument labelled IN (line 5). The iteration continues until no more arguments can be labelled IN or OUT (line 6), and it terminates by labelling UND unlabelled arguments (line 7).

Algorithm 1 Computation of a grounded $\{\text{IN}, \text{OUT}, \text{UND}\}$-labelling.

1: **input** An argumentation graph $\mathcal{G}$,
2: $\text{L}_0 = (\emptyset, \emptyset, \emptyset)$,
3: **repeat**
4: $\quad \text{IN}(\text{L}_{i+1}) \leftarrow \text{IN}(\text{L}_i) \cup \{A | A \in \mathcal{A}_\mathcal{G}$ is not labelled in L_i, and $\forall B \in \mathcal{A}_\mathcal{G}$: if B attacks A then $B \in \text{OUT}(\text{L}_i)\}$
5: $\quad \text{OUT}(\text{L}_{i+1}) \leftarrow \text{OUT}(\text{L}_i) \cup \{A | A \in \mathcal{A}_\mathcal{G}$ is not labelled in L_i, and $\exists B \in \mathcal{A}_\mathcal{G} : B$ attacks A and $B \in \text{IN}(\text{L}_{i+1})\}$
6: **until** $\text{L}_i = \text{L}_{i+1}$
7: **return** $(\text{IN}(\text{L}_i), \text{OUT}(\text{L}_i), \mathcal{A}_\mathcal{G} \backslash (\text{IN}(\text{L}_i) \cup \text{OUT}(\text{L}_i)))$

When some arguments are omitted, we have *grounded* $\{\text{IN}, \text{OUT}, \text{UND}, \text{OFF}\}$-labellings [11], [12], where only expressed arguments can effectively attack other expressed arguments.

Definition 6 (Grounded $\{\text{IN}, \text{OUT}, \text{UND}, \text{OFF}\}$-labelling)**.** *Let $\mathcal{G}$ denote an argumentation graph and $\mathcal{H}$ a subgraph of $\mathcal{G}$. A* grounded $\{\text{IN}, \text{OUT}, \text{UND}, \text{OFF}\}$-labelling *of $\mathcal{G}$ is a* $\{\text{IN}, \text{OUT}, \text{UND}, \text{OFF}\}$*-labelling such that:*

- *every argument in $\mathcal{A}_\mathcal{H}$ is labelled according to the grounded* $\{\text{IN}, \text{OUT}, \text{UND}\}$*-labelling of $\mathcal{H}$,*
- *every argument in $\mathcal{A}_\mathcal{G} \backslash \mathcal{A}_\mathcal{H}$ is labelled* OFF.

An argumentation graph has a unique grounded $\{\text{IN}, \text{OUT}, \text{UND}\}$-labelling, but it has as many grounded $\{\text{IN}, \text{OUT}, \text{UND}, \text{OFF}\}$-labellings as subgraphs.

Up to this point, we have been working with the labelling of arguments with no consideration for the labelling of statements, and in particular for the labelling of the statements supported by arguments. In this regard, we will assume that every argument has one and only one conclusion, which is a statement.

Notation 3. *The conclusion of an argument A is denoted $conc(A)$.*

Given a set of statements, a labelling of this set is a (preferably total) function associating any statement with a label. Amongst possible specifications for labellings of statements [2], we will focus on the bivalent labelling, according to which a statement is either accepted or not, without further sophistication. If a statement is accepted then we will label it 'in', otherwise it is labelled 'no'.

Definition 7 (Bivalent labelling)**.** *Let $\mathcal{G}$ denote an argumentation graph, $\mathcal{L}$ the set of grounded* $\{\text{IN}, \text{OUT}, \text{UND}, \text{OFF}\}$*-labellings of $\mathcal{G}$, Φ a set of statements. A bivalent* $\{\text{in}, \text{no}\}$*-labelling of Φ from* $\text{L} \in \mathcal{L}$ *is a total function* $\text{K} : \mathcal{L}, \Phi \rightarrow \{\text{in}, \text{no}\}$ *such that* $\text{K}(\text{L}, \phi) = \text{in}$ *iff* $\exists A \in \text{IN}(\text{L}) : conc(A) = \phi$.

Notation 4. *A bivalent labelling* K *is represented as a tuple* $\langle \text{in}(\text{K}), \text{no}(\text{K})\rangle$.

To recap, given an argumentation graph, we can label arguments following the specification of grounded

L(B)	IN	IN	IN	IN	OFF	OFF	OFF	OFF
L(C)	OUT	OUT	OFF	OFF	UND	IN	OFF	OFF
L(D)	IN	OFF	IN	OFF	UND	OFF	IN	OFF
K(L.b)	in	in	in	in	no	no	no	no
K(L, c)	no	no	no	no	no	in	no	no
K(L, d)	in	no	in	no	no	no	in	no

Fig. 3. Set of grounded {IN, OUT, UND, OFF}-labellings of the argumentation graph drawn in Figure 1, along with associated bivalent labellings, where b, c and d are the conclusions of the arguments B, C and D, respectively.

{IN, OUT, UND, OFF}-labellings, and from it, we can label a set of statements following the bivalent labelling. On this basis, we are now prepared to better define our learning problem.

III. Learning Problem

From now, we assume a set of bivalent labellings (that can be observed), and we want to find out parsimonious argumentation graphs consistent with every bivalent labelling. Next, we define some terminology to remove possible ambiguities in the remainder of our investigation.

Definition 8 (Consistency)**.**

- *An argumentation graph* $\mathcal{G}$ *and a grounded* {IN, OUT, UND, OFF}*-labelling* L *are consistent iff* L *is a grounded* {IN, OUT, UND, OFF}*-labelling of* $\mathcal{G}$*.*
- *A* {IN, OUT, UND, OFF}*-Labelling* L *and a bivalent labelling* K *are consistent iff* K *is a bivalent labelling from* L*.*
- *An argumentation graph* $\mathcal{G}$ *and a bivalent labelling* K *are consistent iff there exists a grounded* {IN, OUT, UND, OFF}*-labelling* L *of* $\mathcal{G}$ *such that* K *is a bivalent labelling from* L*.*

Definition 9 (Induced argumentation graph)**.** *An argumentation graph* $\mathcal{G}$ *is induced by a set of statements* Φ *iff every statement in* Φ *is the conclusion of at least one argument in* $\mathcal{A}_\mathcal{G}$*.*

We can note that different argument labellings can be consistent with a given statement labelling. For example, and as illustrated in Figure 3, the bivalent labelling $\langle\{\mathsf{b},\mathsf{d}\},\{\mathsf{c}\}\rangle$ can be built from the argument labelling $\mathrm{L}_1 = \langle\{\mathsf{B},\mathsf{D}\},\{\mathsf{C}\},\emptyset,\emptyset\rangle$ or the labelling $\mathrm{L}_2 = \langle\{\mathsf{B},\mathsf{D}\},\emptyset,\emptyset,\{\mathsf{C}\}\rangle$. At its turn, an argument labelling can be consistent with different argumentation graphs. For example, the argument labelling L_2 is the labelling of the graph $\mathcal{G}_1 = \langle\{\mathsf{B},\mathsf{C},\mathsf{D}\},\{\mathsf{B}\rightsquigarrow\mathsf{C},\mathsf{C}\rightsquigarrow\mathsf{D}\}\rangle$ or $\mathcal{G}_2 = \langle\{\mathsf{B},\mathsf{C},\mathsf{D}\},\{\mathsf{B}\rightsquigarrow\mathsf{C}\rangle$, or even $\mathcal{G}_3 = \langle\{\mathsf{B},\mathsf{C},\mathsf{D}\},\{\mathsf{C}\rightsquigarrow\mathsf{B},\mathsf{D}\rightsquigarrow\mathsf{C}\}\rangle$. So, in general, different argumentation graphs can be consistent with the same set of bivalent labellings.

Since different graphs can be consistent with a set of bivalent labellings, we consider Epicurus' Principle of Multiple Explanations according to which if several explanations (let's say argumentation graphs and associated labellings) are consistent with the observed data (here our bivalent labellings) then we should retain them all.

However, Epicurus' principle is not easy to fully satisfy in our context, because the set of graphs consistent with a set of bivalent labellings is infinite. Indeed, and without formalities, we have to look for any argumentation graph $\mathcal{G}$ such that every statement in Φ is the conclusion of at least one argument in $\mathcal{A}_\mathcal{G}$. Consequently, the number of arguments in $\mathcal{G}$ is necessarily superior or equal to the number of observed statements, i.e. $|\Phi| \leq |\mathcal{A}_\mathcal{G}|$. Since the set of arguments $\mathcal{A}_\mathcal{G}$ can be as large as we want, the set of graphs consistent with a set of bivalent labelling is infinite. To address this point, and following Ockham's Principle of Parsimony according to which "Entities must not be multiplied beyond necessity", we may suppose that $|\Phi| = |\mathcal{A}_\mathcal{G}|$, and in this case we say that the argumentation graph $\mathcal{G}$ is an 'argument-parsimonious' argumentation graph induced by Φ.

Definition 10 (Argument-parsimony)**.** *An argumentation graph* $\mathcal{G}$ *is an* argument-parsimonious *argumentation graph induced by a set of statements* Φ *iff* $|\mathcal{A}_\mathcal{G}| = |\Phi|$*.*

Following the Principle of Multiple Explanations and the Principle of Parsimony, we are now prepared to specify our 'abstract argumentation structure learning' problem.

> Abstract argumentation structure learning problem.
>
> Given: a set of observed bivalent labellings of a set of statements Φ,
>
> find (or induce or learn): the set $\mathfrak{H}$ of all the argument-parsimonious argumentation graphs induced by Φ, such that every argumentation graph in $\mathfrak{H}$ is consistent with every observed labelling.

Once the maximal set $\mathfrak{H}$ of argument-parsimonious argumentation graphs consistent with the observed labellings is determined, one may consider the problem of selecting in it a particular graph. This kind of 'selection problem' is thus different from the problem addressed in this paper; we will leave the former to future research.

IV. A Solution By Discarding Attacks

Our problem assumes a set of bivalent labellings of a set of statements Φ, and we have to find out the maximum set $\mathfrak{H}$ of argument-parsimonious argumentation graphs induced by Φ and consistent with every bivalent labelling. To get the set $\mathfrak{H}$, we start from the complete argument-parsimonious argumentation graph $\mathcal{C}$ induced by Φ, i.e. a graph where all the arguments attack each other, and such that every statement in Φ is the conclusion of one and only one argument in $\mathcal{A}_\mathcal{C}$.

Definition 11 (Complete argumentation graph)**.** *A* complete argumentation graph $\mathcal{G}$ *is an argumentation graph such that all the arguments attack each other, i.e.* $\forall A, B \in \mathcal{A}_\mathcal{G}, A \rightsquigarrow_\mathcal{G} B$*.*

Any argument-parsimonious argumentation graphs consistent with the observed labellings of a set of statements Φ is necessarily an 'attack subgraph' (defined below) of the complete argument-parsimonious argumentation graph $\mathcal{C}$ induced by Φ.

Definition 12 (Attack subgraph)**.** *An* attack subgraph $\mathcal{H}$ *of an argumentation graph* $\mathcal{G}$ *is an argumentation graph such that* $\mathcal{A}_\mathcal{H} = \mathcal{A}_\mathcal{G}$ *and* $\rightsquigarrow_\mathcal{H} \subseteq \rightsquigarrow_\mathcal{G}$*.*

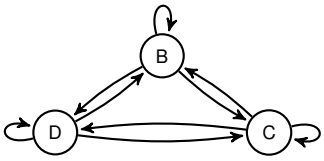


Fig. 4. The complete argument-parsimonious argumentation graph induced by the statements $\{\mathsf{b}, \mathsf{c}, \mathsf{d}\}$.

Example 1. *The graph in Figure 1 is an attack subgraph of the complete graph shown in Figure 4.*

A brute force approach for our problem is to start from the complete argument-parsimonious argumentation graph $\mathcal{C}$ induced by the set of observed statements, and then consider every subgraph of $\mathcal{C}$ along with its bivalent labellings to see which attack subgraphs account for the observed labellings. However this approach is of course not efficient.

As the brute force is not efficient, we investigate here an alternative approach. This alternative consists in approaching our problem by reversing the reasoning typically used in argumentation multi-labelling systems, and this is considered in two stages: in the first stage, from the observation of statement acceptance labellings, we consider $\{\mathsf{IN}, \mathsf{OUT}, \mathsf{UND}, \mathsf{OFF}\}$-labellings consistent with the observed statement labellings, in the second stage, from the $\{\mathsf{IN}, \mathsf{OUT}, \mathsf{UND}, \mathsf{OFF}\}$-labellings induced in the first phase, we revise argument-parsimonious abstract argumentation graphs so that these $\{\mathsf{IN}, \mathsf{OUT}, \mathsf{UND}, \mathsf{OFF}\}$-labellings result grounded. This approach is developed in the remainder of this section.

A. From statement labellings to argument labellings

To induce grounded $\{\mathsf{IN}, \mathsf{OUT}, \mathsf{UND}, \mathsf{OFF}\}$-labellings consistent with an observed bivalent labelling, Definition 7 indicates that, given an argument-parsimonious argumentation graph, any of its grounded $\{\mathsf{IN}, \mathsf{OUT}, \mathsf{UND}, \mathsf{OFF}\}$-labellings consistent with a bivalent labelling must be such that every argument whose conclusion is labelled in must be labelled IN; any other argument is either labelled OUT or UND or OFF. To address this uncertainty, and given a bivalent labelling, we may consider a 'minomit' grounded $\{\mathsf{IN}, \mathsf{OUT}, \mathsf{UND}, \mathsf{OFF}\}$-labelling consistent with it, i.e. a labelling minimising omitted arguments, but it is equally possible to consider a 'maxomit' grounded $\{\mathsf{IN}, \mathsf{OUT}, \mathsf{UND}, \mathsf{OFF}\}$-labelling, i.e. a labelling maximising omitted arguments.

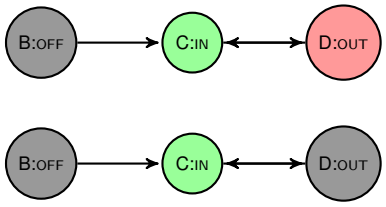


Fig. 5. Minomit (top) and maxomit (bottom) grounded $\{\mathsf{IN}, \mathsf{OUT}, \mathsf{UND}, \mathsf{OFF}\}$-labellings consistent with the bivalent labelling $\langle\{\mathsf{c}\}, \{\mathsf{b}, \mathsf{d}\}\rangle$.

In this paper, we do not have to focus on a particular type of grounded $\{\mathsf{IN}, \mathsf{OUT}, \mathsf{UND}, \mathsf{OFF}\}$-labelling, but whatever the type, it must be consistent with a bivalent labelling. In that regard, grounded $\{\mathsf{IN}, \mathsf{OUT}, \mathsf{UND}, \mathsf{OFF}\}$-labellings are compelling because the computation of such a labelling consistent with a bivalent labelling can be efficiently performed by an adaptation of Algorithm 1, as proposed in Algorithm 2.

Algorithm 2 Computation of a grounded $\{\mathsf{IN}, \mathsf{OUT}, \mathsf{UND}, \mathsf{OFF}\}$-labelling consistent with a bivalent labelling.

1: **input** A bivalent labelling K, an argument-parsimonious argumentation graph $\mathcal{G}$ consistent with K, a set of arguments $\mathsf{OFF}(\mathrm{L}_0) \subseteq \mathcal{A}_\mathcal{G}$ whose conclusions are labelled no in K, i.e. $\forall A \in \mathsf{OFF}(\mathrm{L}_0)$, $conc(A) \in \mathsf{no}(\mathrm{K})$.
2: $\mathrm{L}_0 = (\emptyset, \emptyset, \emptyset, \mathsf{OFF}(\mathrm{L}_0))$.
3: $\mathsf{IN}(\mathrm{L}_0) \leftarrow \{A | A \in \mathcal{A}_\mathcal{G}, conc(A) \in \mathsf{in}(\mathrm{K})\}$.
4: $\mathsf{OUT}(\mathrm{L}_0) \leftarrow \{A | A \in \mathcal{A}_\mathcal{G}$ is not labelled in L_0, $\exists B \in \mathcal{A}_\mathcal{G} : B$ attacks A and $B \in \mathsf{IN}(\mathrm{L}_0)\}$.
5: **repeat**
6: $\quad\mathsf{IN}(\mathrm{L}_{i+1}) \leftarrow \mathsf{IN}(\mathrm{L}_i)$.
7: $\quad\mathsf{OUT}(\mathrm{L}_{i+1}) \leftarrow \mathsf{OUT}(\mathrm{L}_i)$.
8: $\quad\mathsf{OFF}(\mathrm{L}_{i+1}) \leftarrow \mathsf{OFF}(\mathrm{L}_i) \cup \{A | A \in \mathcal{A}_\mathcal{G}$ is not labelled in L_i, and $\forall B \in \mathcal{A}_\mathcal{G} : B$ attacks A and $B \in \mathsf{OUT}(\mathrm{L}_{i+1}) \cup \mathsf{OFF}(\mathrm{L}_i)\}$.
9: **until** $\mathrm{L}_i = \mathrm{L}_{i+1}$
10: $\mathsf{UND}(\mathrm{L}_i) \leftarrow \mathcal{A}_\mathcal{G} \backslash (\mathsf{IN}(\mathrm{L}_0) \cup \mathsf{OUT}(\mathrm{L}_i) \cup \mathsf{OFF}(\mathrm{L}_i))$.
11: **return** $(\mathsf{IN}(\mathrm{L}_0), \mathsf{OUT}(\mathrm{L}_i), \mathsf{UND}(\mathrm{L}_i), \mathsf{OFF}(\mathrm{L}_i))$.

Algorithm 2 begins by labelling IN all the arguments whose conclusions are labelled in (line 3). If an argument is attacked by an argument labelled IN, then it is labelled OUT (line 4). Then an iteration begins. If all the attackers of an argument are labelled OUT or OFF then then this argument is labelled OFF (line 8). The iteration continues until no more arguments can be labelled OFF, and it terminates by labelling UND any argument which remained unlabelled (line 10-11).

The set of arguments labelled IN is minimal since only arguments whose conclusions are labelled in are labelled IN. This minimalisation ensures that the algorithm returns an argument labelling which is grounded and consistent with the input statement labelling. Furthermore, if the input set of arguments labelled OFF is empty, then the output grounded $\{\mathsf{IN}, \mathsf{OUT}, \mathsf{UND}, \mathsf{OFF}\}$-labelling is minomit, while if this input set includes all the arguments whose conclusions are labelled no in K, then the output grounded $\{\mathsf{IN}, \mathsf{OUT}, \mathsf{UND}, \mathsf{OFF}\}$-labelling is maxomit. More generally, the algorithm allows different degrees of argument omission because one can arbitrarily label arguments as OFF in the input (as long as their conclusions are labelled no).

B. From argument labellings to argumentation graphs

If the input argumentation graph in Algorithm 2 is not consistent with the observed bivalent labelling, then the output $\{\mathsf{IN}, \mathsf{OUT}, \mathsf{UND}, \mathsf{OFF}\}$-labelling may not be complete and thus grounded because two arguments labelled IN may attack each other. Hence, to make the input argumentation graph consistent with the observed bivalent labelling, it suffices to discard attacks between arguments labelled IN. By doing so, there always exists

an argument-parsimonious argumentation graph induced by a set of statements which is consistent with any set of bivalent labellings of these statements.

Theorem 1 (Consistent Argumentation Graph)**.** *Let $\mathcal{K}$ denote a set of bivalent labellings of a set of statements Φ. An argument-parsimonious argumentation graph $\mathcal{G}$ induced by Φ is consistent with every labelling in $\mathcal{K}$ iff $\forall A, B \in \mathcal{A}_\mathcal{G}$: if $\exists \mathrm{K} \in \mathcal{K}$ $conc(A), conc(B) \in \mathsf{in}(\mathrm{K})$ then $A \not\rightsquigarrow_\mathcal{G} B$.*

Proof. ($\Rightarrow$) If an argument-parsimonious argumentation graph $\mathcal{G}$ is consistent with every labelling in $\mathcal{K}$ then $\forall A, B \in \mathcal{A}_\mathcal{G}$: if $\exists \mathrm{K}(\mathrm{L}, \cdot) \in \mathcal{K}$ $conc(A), conc(B) \in \mathsf{in}(\mathrm{K})$ then $\mathrm{L}(A) = \mathrm{L}(B) = \mathsf{IN}$, and thus $A \not\rightsquigarrow_\mathcal{G} B$. ($\Leftarrow$) Let $\mathcal{G}$ denote an argument-parsimonious argumentation graph induced by Φ such that $\forall A, B \in \mathcal{A}_\mathcal{G}$: if $\exists \mathrm{K} \in \mathcal{K}$ $conc(A), conc(B) \in \mathsf{in}(\mathrm{K})$ then $A \not\rightsquigarrow_\mathcal{G} B$. For every K in $\mathcal{K}$, the $\{\mathsf{IN}, \mathsf{OUT}, \mathsf{UND}, \mathsf{OFF}\}$-labelling computed using Algorithm 2, with $\mathcal{G}$ and K as inputs, is a grounded $\{\mathsf{IN}, \mathsf{OUT}, \mathsf{UND}, \mathsf{OFF}\}$-labelling of $\mathcal{G}$ consistent with K. Therefore, $\mathcal{G}$ is consistent with every labelling in $\mathcal{K}$. □

Theorem 2 (Existence)**.** *Let $\mathcal{K}$ denote a set of bivalent labellings of a set of statements Φ. There exists an argument-parsimonious argumentation graph $\mathcal{G}$ induced by Φ, which is consistent with every* K *in $\mathcal{K}$.*

Proof. Trivial since it suffices to consider an argumentation graph $\mathcal{G}$ as defined in Theorem 1. □

More interestingly, given a set of bivalent labellings, it is also straightforward to consider an argument-parsimonious argumentation graph whose attack subgraphs are the maximal set of graphs consistent with every bivalent labelling.

Theorem 3 (Maximal set)**.** *Let $\mathcal{K}$ be a set of bivalent labellings of a set of statements Φ, $\mathcal{G}$ the argument-parsimonious argumentation graph induced by Φ such that $\forall A, B \in \mathcal{A}_\mathcal{G}$: if $\exists \mathrm{K} \in \mathcal{K}$ $conc(A), conc(B) \in \mathsf{in}(\mathrm{K})$ then $A \not\rightsquigarrow_\mathcal{G} B$ else $A \rightsquigarrow_\mathcal{G} B$. The set of attack subgraphs of $\mathcal{G}$ is the maximal set of argument-parsimonious argumentation graphs consistent with every labelling in $\mathcal{K}$.*

Proof. By contradiction. If the set $\mathfrak{G}$ of attack subgraphs of $\mathcal{G}$ is not maximal, then there is an argument-parsimonious argumentation graph $\mathcal{X}$ consistent with $\mathcal{K}$ which is not included in it. However, since $\mathcal{X}$ is an argument-parsimonious argumentation graph consistent with $\mathcal{K}$, $\forall A, B \in \mathcal{A}_\mathcal{X}$: if $\exists K \in \mathcal{K}$ $conc(A), conc(B) \in \mathsf{in}(\mathrm{K})$ then $A \not\rightsquigarrow_\mathcal{X} B$ (Theorem 1). Consequently, $\mathcal{X}$ is an attack subgraph of $\mathcal{G}$, and thus $\mathcal{X}$ is included in $\mathfrak{G}$, which contradicts that $\mathcal{X}$ is not included in $\mathfrak{G}$. □

Thanks to grounded $\{\mathsf{IN}, \mathsf{OUT}, \mathsf{UND}, \mathsf{OFF}\}$-labellings, theorems 1 and 3 indicate us a simple approach to ensure the induction of an argument-parsimonious argumentation graph whose attack subgraphs are the maximal set of graphs consistent with a set of bivalent labellings: we just have to discard attacks amongst arguments whose conclusions are labelled in in any bivalent labelling; and we will consider that simple idea next.

C. Abstract argumentation structure learning algorithm

We are now prepared to propose an algorithm (Algorithm 3) to solve our abstract argumentation structure learning problem: it iteratively discards attacks of the complete argument-parsimonious argumentation graph induced by a set of statements until the set of attacks is empty or some predefined computational budget (typically time) is reached, at which point the loop is halted and the argumentation graph induced so far is returned.

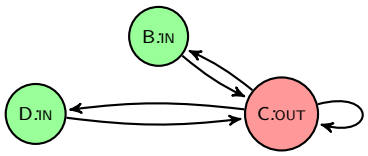
Algorithm 3 Abstract argumentation structure learning algorithm.

1: **input** The complete argument-parsimonious argumentation graph $\mathcal{X}$ induced by a set of statements Φ.
2: **while** $\rightsquigarrow_\mathcal{X} \neq \emptyset$ or within computational budget **do**
3: Get a bivalent labelling K of Φ.
4: **if** $\exists A, B \in \mathcal{A}_\mathcal{X}$ such that $conc(A), conc(B) \in \mathsf{in}(\mathrm{K})$ **then**
5: $\rightsquigarrow_\mathcal{X} \leftarrow \rightsquigarrow_\mathcal{X} \setminus \{(A, B), (B, A)\}$.
6: **end if**
7: **end while**
8: **return** the argumentation graph $\mathcal{X}$.

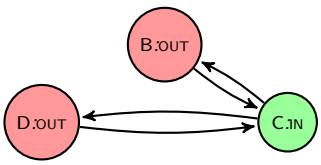
The algorithm starts from a complete argument-parsimonious argumentation graph $\mathcal{X}$ induced by a set of statements (line 1) whose some bivalent labellings will be observed. Every time that a bivalent labelling is observed (line 3), if there exist arguments whose conclusions are labelled in, then the mutual attacks between these arguments are discarded (line 5), until some computational budget is reached or the set of attacks is empty. When the the computational budget is reached, the induced argumentation graph is returned.

Example 2. *Let us illustrate Algorithm 3. Consider the argumentation graph shown in Figure 1 as the source of a stream of bivalent labellings that we will observe. The input argumentation graph of Algorithm 3 is pictured in Figure 4.*

1st observed labelling is $\langle\{\mathsf{b}, \mathsf{d}\}, \{\mathsf{c}\}\rangle$. The revised argument argumentation consistent with it is as follows (its minomit grounded $\{\mathsf{IN}, \mathsf{OUT}, \mathsf{UND}, \mathsf{OFF}\}$-labelling is shown for illustrative purposes, but any grounded $\{\mathsf{IN}, \mathsf{OUT}, \mathsf{UND}, \mathsf{OFF}\}$-labelling computed with Algorithm 2 would do the illustration as well).



2nd observed labelling is $\langle\{\mathsf{c}\}\}, \{\mathsf{b}, \mathsf{d}\}\rangle$. Here the revised argumentation graph consistent with it:



We can see that all attack subgraphs of the above argumentation graph (including the graph in Figure 1) are consistent with every statement labelling observed from the beginning.

Given a set of bivalent labellings observed by Algorithm 3 (in Line 3), Algorithm 3 returns an argumentation graph whose every attack subgraph is consistent with the observed labellings. In that regard, Algorithm 3 is sound.

Theorem 4 (Soundness)**.** *Algorithm 3 is sound.*

Proof. Let $\mathcal{K}$ be a set of bivalent labellings of a set of statements Φ such that every labelling in this set is observed by Algorithm 3. Algorithm 3 returns an argument-parsimonious argumentation graph $\mathcal{G}$ induced by Φ such that $\forall A, B \in \mathcal{A}_\mathcal{G}$ if $\exists \mathrm{K} \in \mathcal{K}$ $conc(A), conc(B) \in \mathsf{in}(\mathrm{K})$, we have $A \not\rightsquigarrow_\mathcal{G} B$. By theorem 1, the argumentation graph $\mathcal{G}$ and all its attack subgraphs are thus consistent with every K in $\mathcal{K}$. □

Moreover, Algorithm 3 returns an argumentation graph whose set of all attack subgraphs is the maximal set of argument parsimonious argumentation graphs consistent with the set of observed labellings. In that regard, Algorithm 3 is complete.

Theorem 5 (Completeness)**.** *Algorithm 3 is complete.*

Proof. Let $\mathcal{K}$ be a set of bivalent labellings of a set of statements Φ such that every labelling in this set is observed by Algorithm 3. Algorithm 3 returns an argument-parsimonious argumentation graph $\mathcal{G}$ induced by Φ such that $\forall A, B \in \mathcal{A}_\mathcal{G}$: if $\exists \mathrm{K} \in \mathcal{K}$ $conc(A), conc(B) \in \mathrm{K}(\mathsf{in})$, we have $A \not\rightsquigarrow_\mathcal{G} B$ else $A \rightsquigarrow_\mathcal{G} B$. Therefore, by Theorem 3, the set of all attack subgraphs of $\mathcal{G}$ is the maximal set of argumentation graphs consistent with the set of observed labellings. □

As Example 2 illustrates, Algorithm 3 works 'on the fly' and thus, as a major benefit, the observed bivalent labellings do not have to be kept in memory. However we do have to keep in memory all the possible attacks between arguments amongst $|\mathcal{A}_\mathcal{G}|$ arguments of a complete argument-parsimonious argumentation graph $\mathcal{G}$ induced by a set of statements Φ, with $|\mathcal{A}_\mathcal{G}| = |\Phi|$. Therefore, the computation can be performed in a space that is quadratic in the number of statements.

Theorem 6 (Space complexity)**.** *Given a set of statements Φ whose bivalent labellings are observed in Algorithm 3, the computation of Algorithm 3 is $O(|\Phi|^2)$.*

Concerning time complexity, the 'burden' stems from the elimination of attacks in line 5 of Algorithm 3, something achievable in linear time. Thus every iteration can be performed in a time that is quadratic in the number of statements.

Theorem 7 (Time complexity)**.** *Given a set of statements Φ whose bivalent labellings are observed in Algorithm 3, an iteration step in Algorithm 3 is $O(|\Phi|^2)$.*

Given the above theorems, we omit experimental insights because Algorithm 3 will always return an argumentation graph whose attack subgraphs are the maximum set of argument-parsimonious argumentation graphs consistent with every observed bivalent statement labelling.

V. Conclusion

We settled an abstract structure learning problem in abstract argumentation: bivalent labellings of statements are observed and, in accordance with Epicurus' Principle of Multiple Explanations and Ockham's Principle of Parsimony, the goal is to find out all argument-parsimonious argumentation graphs consistent with the observed labellings.

To address this problem, we inversed the typical reasoning in multi-labelling systems: from the observation of statement acceptance labellings, we (virtually) consider argument acceptance labellings consistent with the statement labellings, and from the argument acceptance labellings, the underlying argumentation graph is updated to make it consistent with the labellings.

Our approach is based on grounded $\{\mathsf{IN}, \mathsf{OUT}, \mathsf{UND}, \mathsf{OFF}\}$ labellings for arguments and associated bivalent labellings for statements, and thanks to these labellings, we are able to propose a simple anytime and 'on the fly' algorithm (Algorithm 3) taking as input a stream of bivalent labellings, revising an argumentation graph, and returning an argumentation graph whose set of attack subgraphs is the maximal set of argumentation graphs consistent with the observed statement labellings. We showed that the proposed algorithm is efficient.

References

[1] P. Baroni, M. Caminada, and M. Giacomin. An introduction to argumentation semantics. *Knowledge Engineering Review*, 26(4):365–410, 2011.

[2] P. Baroni, G. Governatori, and R. Riveret. On labelling statements in multi-labelling argumentation. In *Proc. of the 22nd Eur. Conf. on Artificial Intelligence*, pages 489–497. IOS Press, 2016.

[3] R. Baumann. What does it take to enforce an argument? minimal change in abstract argumentation. In *Proc. of the 20th Eur. Conf. on Artificial Intelligence*, pages 127–132. IOS Press, 2012.

[4] T. Bench-Capon and P. Dunne. Argumentation in artificial intelligence. *Artificial Intelligence*, 171(10):619 – 641, 2007.

[5] P. M. Dung. On the acceptability of arguments and its fundamental role in nonmonotonic reasoning, logic programming and n-person games. *Artif. Intell.*, 77(2):321–358, 1995.

[6] P. E. Dunne, W. Dvorák, T. Linsbichler, and S. Woltran. Characteristics of multiple viewpoints in abstract argumentation. *Artif. Intell.*, 228:153–178, 2015.

[7] S. Modgil and M. Caminada. *Proof Theories and Algorithms for Abstract Argumentation Frameworks*, pages 105–129. Springer, 2009.

[8] A. Niskanen, J. P. Wallner, and M. Jrvisalo. Synthesizing argumentation frameworks from examples. In *Proc. of the 22nd Eur. Conf. on Artificial Intelligence*, pages 551–559. IOS Press, 2016.

[9] I. Rahwan and G. R. Simari. *Argumentation in Artificial Intelligence*. Springer, 1st edition, 2009.

[10] R. Riveret and G. Governatori. On learning attacks in probabilistic abstract argumentation. In *Proc. of the 15th Int. Conf. on Autonomous Agents & Multiagent Systems*, pages 653–661. IFAAMAS, 2016.

[11] R. Riveret, D. Korkinof, M. Draief, and J. Pitt. Probabilistic abstract argumentation: an investigation with boltzmann machines. *Argument & Computation*, 6(2):178–218, 2015.

[12] R. Riveret, J. Pitt, D. Korkinof, and M. Draief. Neuro-symbolic agents: Boltzmann machine and Probabilistic Abstract Argumentation with Sub-arguments. In *Proc. of the 14th Int. Conf. on Autonomous Agents & Multiagent Systems*, pages 1481–1489. IFAAMAS, 2015.