Complexity Classifications for logic-based Argumentation*

NADIA CREIGNOU
Aix-Marseille Université

UWE EGLY†
Technische Universität Wien

JOHANNES SCHMIDT
Linköping University

April 22, 2013

Abstract

We consider logic-based argumentation in which an argument is a pair (Φ, α), where the support Φ is a minimal consistent set of formulæ taken from a given knowledge base (usually denoted by Δ) that entails the claim α (a formula). We study the complexity of three central problems in argumentation: the existence of a support $\Phi \subseteq \Delta$, the validity of a support and the relevance problem (given ψ is there a support Φ such that $\psi \in \Phi$?). When arguments are given in the full language of propositional logic these problems are computationally costly tasks, the validity problem is DP-complete, the others are Σ_2^{p}-complete. We study these problems in Schaefer's famous framework where the considered propositional formulæ are in generalized conjunctive normal form. This means that formulæ are conjunctions of constraints build upon a fixed finite set of Boolean relations Γ (the constraint language). We show that according to the properties of this language Γ, deciding whether there exists a support for a claim in a given knowledge base is either polynomial, NP-complete, coNP-complete or Σ_2^{p}-complete. We present a dichotomous classification, P or DP-complete, for the verification problem and a trichotomous classification for the relevance problem into either polynomial, NP-complete, or Σ_2^{p}-complete. These last two classifications are obtained by means of algebraic tools.

1 Introduction

Argumentation can be seen as a generalization of many forms of nonmonotonic reasoning previously developed [Dun95]. It is nowadays a very active research area in artificial intelligence. One can identify, among others, two important lines of research: *abstract* argumentation [Dun95] and *logic-based* (or deductive) argumentation [BH01, CML00, PV02, BH08]. The former focuses on the relations between arguments based on the property of arguments to attack

*A preliminary version of this work appeared in the Proceedings of COMMA 2012, *Frontiers in Artificial Intelligence and Applications*, Volume 245, pages 237–248, 2012.

†This work was partially supported by the Austrian Science Foundation (FWF) under grant S11409-N23

others, thereby ignoring the internal structure of an argument and the nature of the attack relation. In this work we explore logic-based argumentation in which an argument is a pair (Φ, α), where the support Φ is a minimal consistent set of formulæ that entails the claim α (a formula).

From a complexity theoretic viewpoint, computing the support of an argument is a very hard problem. Indeed, in the full language of propositional logic, given a knowledge base Δ, the problem of deciding whether there exists a support $\Phi \subseteq \Delta$ for a given claim α has been shown to be Σ_2^{p}-complete [PWA03]. Since this problem underlies many reasoning problems in logic-based argumentation, like for instance the computation of argument trees as proposed by Besnard and Hunter [BH01], it is natural to try to identify fragments of propositional logic for which the deduction problem is easier.

A first step towards an extensive study of the complexity of argumentation in fragments of propositional logic was taken in [CSTW11] in Post's framework, where the authors considered formulæ built upon a restricted set of connectives. They obtained a full classification of various argumentation problems depending on the set of allowed connectives. A similar yet different approach is not to restrict the connectives but to restrict the syntactic shape of the formulæ. This refers to the well-known Schaefer's framework in which formulæ are considered in generalized conjunctive normal form with clauses formed upon a fixed set of relations Γ (the constraint language). Such formulæ are called Γ-*formulæ*. This framework captures well-known classes of formulæ in conjunctive normal form, e.g., Horn, definite Horn or 2-CNF. A wide range of algorithmic problems have been studied in this context (for a survey see [CV08]), and in particular the abduction problem [CZ06, NZ08].

Our main contribution is a systematic complexity classification for the problems of existence Arg, validity Arg-Check and relevance Arg-Rel in terms of all possible sets of relations Γ. These problems are formally defined in Section 3. They can be described as follows:

- Arg: given (Δ, α), does there exist a support $\Phi \subseteq \Delta$ for α ?
- Arg-Check: given (Φ, α), is it an argument?
- Arg-Rel: given (Δ, α, ψ), is there a support $\Phi \subseteq \Delta$ such that $\psi \in \Phi$?

We prove that depending on the set of allowed relations Γ in our formulæ in generalized conjunctive normal form, deciding the existence of a support is either in P, or NP-complete, or coNP-complete or Σ_2^{p}-complete. The validity problem Arg-Check is either in P or DP-complete, whereas the relevance problem Arg-Rel obtains a trichotomous classification into membership in P, or NP-complete, or Σ_2^{p}-complete.

For many classifications obtained in Schaefer's framework the so-called *algebraic approach* turned out to be applicable. This means that the complexity of a problem parametrized by a constraint language Γ is fully determined by the relational clone $\langle\Gamma\rangle$, the closure of Γ under primitive positive definitions. In the case of the argumentation problems we consider, it is however not clear how to prove such a statement on the complexity. We therefore develop some new techniques that still allow us to use parts of these elegant algebraic tools. While in the case of Arg and Arg-Check we finally obtain that their complexity does not change within relational clones, we show that in the case of Arg-Rel the usual algebraic approach is definitely not applicable (unless P = NP): we identify constraint languages Γ_1, Γ_2 that give rise to the same relational clone, i.e., $\langle\Gamma_1\rangle = \langle\Gamma_2\rangle$, such that Arg-Rel($\Gamma_1$) is in P and Arg-Rel($\Gamma_2$) is NP-complete.

The paper is organized as follows. In Section 2 we give some basics on complexity theory, we present Schaefer's framework and we remind some complexity classifications that will be of use in our proofs (in particular we explain how our work relates to the complexity classifications obtained for abduction). In Section 3 we define formally the problems we are interested in. In Section 4 we present the algebraic tools we will use and give a series of technical lemmas. In the following sections we establish complexity classifications for the existence (Section 5), validity (Section 6) and relevance (Section 7) problems. We conclude in Section 8.

2 Preliminaries

We assume familiarity with the syntax and semantics of propositional and first order logic. A *literal* is a variable (*positive literal*) or its negation (*negative literal*), a (k-) *clause* is a disjunction of (k) literals and a formula in (k-) *CNF* is a conjunction of (k-) clauses. A formula in CNF is *Horn* (resp., *dual Horn*) if every clause contains at most one positive (resp., negative) literal. A formula in CNF is *positive* (resp., *negative*) if every clause contains positive (resp., negative) literals only.

2.1 Complexity theory

We require standard notions of complexity theory. For the problems studied in the paper the arising complexity degrees encompass the classes P, NP, coNP, DP and Σ_2^{p}, where DP is defined as the set of languages recognizable by the difference of two languages in NP, i.e., $\mathrm{DP} := \{L_1 \setminus L_2 \mid L_1, L_2 \in \mathrm{NP}\} = \{L_1 \cap L_2 \mid L_1 \in \mathrm{NP}, L_2 \in \mathrm{coNP}\}$, and Σ_2^{p} is the set of languages recognizable by nondeterministic polynomial-time Turing machines with an NP oracle. For our hardness results we employ *logspace many-one reductions*, defined as follows: a language A is logspace many-one reducible to some language B (written $A \leq_{\mathrm{m}}^{\log} B$) if there exists a logspace-computable function f such that $x \in A$ if and only if $f(x) \in B$. For more background information on complexity theory, the reader is referred to [Pap94]. We will use, among others, the following standard problems to prove hardness results.

Problem: 3-Sat (NP-complete according to [Coo71])
Instance: A propositional formula φ in 3-CNF.
Question: Is φ satisfiable?

Problem: Pos-1-in-3-Sat (NP-complete according to [Sch78])
Instance: A propositional formula φ in 3-CNF with only positive literals.
Question: Is there an assignment to the variables of φ that sets in each clause exactly one variable to true?

Problem: Critical-Sat (DP-complete according to [PW88])
Instance: A propositional formula φ in 3-CNF.
Question: Is φ unsatisfiable but removing any of its clauses makes it satisfiable?

2.2 Constraint languages and Γ-formulæ

A *logical relation* of arity k is a relation $R \subseteq \{0,1\}^k$. In this paper we will only consider nontrivial relations, i.e., $R \neq \emptyset$ and $R \neq \{0,1\}^k$. By abuse of notation we do not make a

difference between a relation and its predicate symbol. We will use T and F as the two unary constant relations $\mathrm{T} = \{1\}$ and $\mathrm{F} = \{0\}$. A *constraint*, C, is a formula $C = R(x_1, \ldots, x_k)$, where R is a logical relation of arity k and the x_i's are (not necessarily distinct) variables. For instance the two constraints $\mathrm{T}(x)$ and $\mathrm{F}(x)$ stand for the two unit clauses (x) and $(\neg x)$, respectively. If u and v are two variables, then $C[v/u]$ denotes the constraint obtained from C by replacing each occurrence of v by u. If V is a set of variables, then $C[V/u]$ denotes the result of substituting u to every occurrence of every variable of V in C. An assignment m of truth values to the variables *satisfies* the constraint C if $\big(m(x_1), \ldots, m(x_k)\big) \in R$. A *constraint language* Γ is a finite set of nontrivial logical relations. A Γ-*formula* ϕ is a conjunction of constraints using only logical relations from Γ and is hence a quantifier-free first-order formula. With $\mathrm{var}(\phi)$ we denote the set of (free) variables appearing in ϕ. A Γ-formula ϕ is satisfied by an assignment $m : \mathrm{var}(\phi) \to \{0, 1\}$ if m satisfies all constraints in ϕ simultaneously (such a satisfying assignment is also called a *model* of ϕ). Assuming a canonical order on the variables we can regard models as tuples in the obvious way and we do not distinguish between a formula ϕ and the logical relation R_ϕ it defines, i.e., the relation consisting of all models of ϕ. We say that two first-order formulæ φ and ψ are equivalent, $\varphi \equiv \psi$, if every assignment $m : \mathrm{var}(\varphi) \cup \mathrm{var}(\psi) \to \{0, 1\}$ on the combined variable sets satisfies φ if and only if it satisfies ψ. We write $\varphi \models \psi$ if φ entails ψ, i.e., if ψ is satisfied by any assignment $m : \mathrm{var}(\varphi) \cup \mathrm{var}(\psi) \to \{0, 1\}$ that satisfies φ.

Throughout the text we refer to different types of Boolean relations following Schaefer's terminology [Sch78]. We say that a Boolean relation R is

- *Horn* (resp. *dualHorn*) if R can be defined by a CNF formula which is Horn (resp. dualHorn);
- *bijunctive* if it can be defined by a 2-CNF formula;
- *affine* if it can be defined by an affine formula, i.e., a conjunction of XOR-clauses (consisting of an XOR of some variables plus maybe the constant 1) — such a formula may also be seen as a system of linear equations over GF[2];
- *positive* (resp. *negative*) if R can be defined by a positive (resp. negative) CNF formula;
- *0-valid* (resp., *1-valid*) if $R(0, \ldots, 0) = 1$ (resp., $R(1, \ldots, 1) = 1$);
- *ε-valid* if R is either 0-valid, or 1-valid or both;
- *complementive* if, for all $m \in R$, we have also $\overline{m} \in R$, where $\overline{m}$ denotes the dual assignment of m defined by $\overline{m}(x) = 1 - m(x)$.

Finally a constraint language Γ is Horn (resp. dualHorn, bijunctive, affine, positive, negative, 0-valid, 1-valid, ϵ-valid, complementive) if every relation in Γ is Horn (resp. dualHorn, bijunctive, affine, positive, negative, 0-valid, 1-valid, ϵ-valid, complementive). We say that a constraint language is *Schaefer* if Γ is either Horn, dualHorn, bijunctive, or affine.

There exist easy criteria to determine if a given relation is Horn, dualHorn, bijunctive, or affine. Indeed all these classes can be characterized by their polymorphisms (see e.g., [CV08] for a detailed description). We recall here the characterizations for Horn and dualHorn. The binary operations of conjunction and disjunction applied on k-ary Boolean vectors are applied coordinate-wise.

- R is Horn if and only if $m, m' \in R$ implies $m \wedge m' \in R$.
- R is dualHorn if and only if $m, m' \in R$ implies $m \vee m' \in R$.

2.3 Related complexity classifications

The formulæ in generalized conjunctive normal form, Γ-formulæ, defined as in the section above, have provided a rich framework to obtain complexity classifications for computational problems involving Boolean formulæ (see e.g., [CV08]). We recall here some of them that will be of use in the following. Moreover we make clear the relationship between the complexity of argumentation and the complexity of abduction.

The satisfiability problem for Γ-formulæ, denoted by $\textsc{Sat}(\Gamma)$, was first studied by Schaefer [Sch78] who obtained a famous dichotomous classification: If Γ is Schaefer or 0-valid or 1-valid, then $\textsc{Sat}(\Gamma)$ is in P; otherwise $\textsc{Sat}(\Gamma)$ is NP-complete.

The complexity of the implication problem for Γ-formulæ was studied in [SS08]. The authors obtain a dichotomous classification for $\textsc{Imp}(\Gamma)$ (i.e., given φ and ψ two Γ-formulæ, does $\varphi \models \psi$ hold?): it is in P if Γ is Schaefer and coNP-complete otherwise.

Since then and in the recent past, complexity classifications for many further computational problems for Γ-formulæ have been obtained (see [CV08] for a survey). In particular we will consider the following abduction problems.

Problem:	$\textsc{Abd}(\Gamma)$.
Instance:	$\mathcal{A} = (\varphi, H, q)$, where φ is a Γ-formula, H is set of variables, and $q \notin H$ is a variable.
Question:	Does there exist $E \subseteq \mathrm{Lits}(H)$ (where $\mathrm{Lits}(H)$ denotes the set of literals that can be built upon variables from H) such that $\varphi \wedge E$ is satisfiable and $\varphi \wedge E \models q$?

Problem:	P-$\textsc{Abd}(\Gamma)$.
Instance:	$\mathcal{A} = (\varphi, H, q)$, where φ is a Γ-formula, H is set of variables, and $q \notin H$ is a variable.
Question:	Does there exist $E \subseteq H$ such that $\varphi \wedge E$ is satisfiable and $\varphi \wedge E \models q$?

According to the classifications obtained in [CZ06, NZ08] we will use the fact that if Γ is not Schaefer, then $\textsc{Abd}(\Gamma)$ is Σ_2^{p}-complete and that if Γ is in addition neither 0-valid, nor 1-valid then P-$\textsc{Abd}(\Gamma)$ is Σ_2^{p}-complete, too.

We want to outline at this point the seeming proximity of argumentation to abduction. In full propositional logic the abduction problem and the argumentation problem are equivalent (with respect to polynomial many-one reductions) since they are both complete for the second level of the polynomial hierarchy ([EG95, PWA03]). Indeed there are very simple reductions proving this equivalence. We give here exemplary the reductions between P-$\textsc{Abd}$ and $\textsc{Arg}$.

1. P-$\textsc{Abd} \leq_{\mathrm{m}}^{\log} \textsc{Arg}$: $(\varphi, H, q) \mapsto (\Delta, \alpha)$, where
   $\Delta := \{\varphi\} \cup H$, $\alpha := q \wedge \varphi$.
2. $\textsc{Arg} \leq_{\mathrm{m}}^{\log}$ P-$\textsc{Abd}$: $(\Delta, \alpha) \mapsto (\varphi, H, q)$, where
   $\Delta = \{\varphi_1, \ldots, \varphi_n\}$, $H := \{x_1, \ldots, x_n\}$ where the $x_i's$ are fresh variables,
   $\varphi := (\alpha \leftrightarrow q) \wedge \bigwedge_{i=1}^{n} (x_i \leftrightarrow \varphi_i)$.

For fragments of propositional logic these reductions do generally not preserve the properties of the chosen fragment and are thus not suited to transfer complete complexity classifications between abduction and argumentation. Nevertheless we will use the idea of the first reduction to transfer certain hardness results from abduction to argumentation. For instance by the first reduction and hardness results in [NZ08] one obtains immediately that deciding the existence of a support for Horn-formulæ is NP-hard. Since for Horn-formulæ satisfiability and implication are in P, the verification problem in comparison is in P.

3 Argumentation problems

In this section we define the computational problems we are interested in.

Definition 3.1. [BH01] *An* argument *is a pair* (Φ, α)*, where* Φ *is a set of formulæ and* α *is a formula such that*

1. Φ *is consistent,*
2. $\Phi \models \alpha$*,*
3. Φ *is minimal with property (2),* i.e.*, no proper subset of* Φ *entails* α*.*

We say that (Φ, α) *is an* argument *for* α*. If* $\Phi \subseteq \Delta$ *then it is said to be an* argument in Δ*. We call* α *the* claim *and* Φ *the* support *of the argument.*

Note that in a more general setting a support Φ for a claim α is a set of formulæ such that Φ is consistent and $\Phi \models \alpha$ and no minimality is required. However, in the definition of an argument, the support is a minimal one.

Let Γ be a constraint language. Then the *argument existence problem for* Γ*-formulæ* is defined as follows:

Problem:	ARG(Γ).
Instance:	(Δ, α), where Δ is a set of Γ-formulæ and α is a Γ-formula.
Question:	Does there exist Φ such that (Φ, α) is an argument in Δ?

Besides the decision problem for the existence of an argument we are interested in the verification problem ARG-CHECK(Γ) and in the relevance problem ARG-REL(Γ), which are defined as follows:

Problem:	ARG-CHECK(Γ).
Instance:	(Φ, α), where Φ is a set of Γ-formulæ and α is a Γ-formula.
Question:	Is (Φ, α) an argument?

Problem:	ARG-REL(Γ).
Instance:	(Δ, α, ψ), where Δ is a set of Γ-formulæ, $\psi \in \Delta$ and α is a Γ-formula.
Question:	Does there exist Φ such that $\psi \in \Phi$ and (Φ, α) is an argument in Δ ?

4 Methods and technical tools

4.1 Co-clones and Galois connection

We now introduce the logical and algebraic tools that our hardness proofs rely on. For establishing the complexity of the argumentation problems when restricted to Γ-formulæ, the key will be to study the expressive power of the set Γ. This expressivity can be more or less restricted as discussed in the following definition where the notations from [SS08] are adopted.

Definition 4.1. *Let Γ be a constraint language.*

- *The set $\langle\Gamma\rangle$ is the smallest set of relations that contains Γ and the equality constraint, $=$, and which is closed under primitive positive first order definitions, i.e., if ϕ is a $\Gamma \cup \{=\}$-formula and $R(x_1, \ldots, x_n) \equiv \exists y_1 \ldots \exists y_l \phi(x_1, \ldots, x_n, y_1, \ldots, y_l)$, then $R \in \langle\Gamma\rangle$. In other words, $\langle\Gamma\rangle$ is the set of relations that can be expressed as a $\Gamma \cup \{=\}$-formula with existentially quantified variables.*
- *The set $\langle\Gamma\rangle_{\neq}$ is the set of relations that can be expressed as a Γ-formula with existentially quantified variables (no equality relation is allowed).*
- *The set $\langle\Gamma\rangle_{\not\exists,\neq}$ is the set of relations that can be expressed as a Γ-formula (neither equality relation nor existential quantification is allowed).*

Let us explain why these closure operators are relevant for us. Assume that $\Gamma_1 \subseteq \langle\Gamma_2\rangle_{\not\exists,\neq}$. Then any Γ_1-formula can be transformed into an equivalent Γ_2-formula. Since for such equivalent formulas the answers to the problems that we consider in this paper are the same, the closure operator $\langle . \rangle_{\not\exists,\neq}$ directly induces reductions for our problems, e.g., $\textsc{Arg}(\Gamma_1) \leq_{\mathrm{m}}^{\log} \textsc{Arg}(\Gamma_2)$.

This notion of expressibility can be relaxed in allowing equality relations and existential quantification. For some computational problems this is still relevant. For instance, assume that $\Gamma_1 \subseteq \langle\Gamma_2\rangle$. Then we have a procedure to transform any Γ_1-formula into a satisfiability-equivalent Γ_2-formula: the equivalent Γ_2-formula contains additional existentially quantified first-order variables and equality constraints can occur. The existential quantifiers can be removed and the equality constraints can be dealt with by identification of variables. Thus, it has been shown that $\textsc{Sat}(\Gamma_1)$ can be reduced in logarithmic space to $\textsc{Sat}(\Gamma_2)$ (see [Jea98, ABI$^+$05]). Hence, the complexity of $\textsc{Sat}(\Gamma)$ depends only on $\langle\Gamma\rangle$. The set $\langle\Gamma\rangle$ is called *relational clone* (or a *co-clone*). Accordingly, in order to obtain a full complexity classification for the satisfiability problem one only has to study the co-clones.

Interestingly, there exists a Galois correspondence between the lattice of Boolean relations (co-clones) and the lattice of Boolean functions (clones) (see [Gei68, BKKR69]). As a consequence, based on the famous Post's description of the lattice of clones [Pos41], the lattice of co-clones is nowadays well-known (see e.g., [BRSV05, CKZ08]). Therefore this Galois connection and this lattice provide a very powerful tool that can be successfully applied in order to obtain complexity classifications for computational problems dealing with Boolean formulæ (see e.g., [CV08] for a survey and [NZ08] for certain variants of the abduction problem).

However, this Galois connection is apparently not appropriate in order to transfer complexity results in the case of argumentation. Indeed, suppose that $\varphi(\overline{x})$ is logically equivalent to $\exists\overline{y}\varphi'(\overline{x},\overline{y})$. It is clear that φ is satisfiable if and only if φ' is satisfiable. Moreover, for any formula $\psi(\overline{x})$ we have that $\varphi \models \psi$ if and only if $\varphi' \models \psi$. However, if $\psi(\overline{x})$ itself is logically

equivalent to $\exists \overline{u}\psi'(\overline{x}, \overline{u})$, it is not true any more that $\varphi \models \psi$ implies $\varphi \models \psi'$ (and neither $\varphi' \models \psi'$). Therefore, when transforming instances between argumentation problems, introducing existential variables is problematic with respect to the claim. For this reason we will introduce a technical version of the two problems Arg-Check and Arg-Rel in which we can differentiate the restrictions put on the knowledge base from the ones put on the claim. The variants we will use are defined as follows.

Problem: Arg-Check(Γ, R).
Instance: (Φ, α), where Φ is a set of Γ-formulæ and α is an R-constraint.
Question: Is (Φ, α) an argument?

Problem: Arg-Rel(Γ, R).
Instance: (Δ, α, ψ), where Δ is a set of Γ-formulæ, $\psi \in \Delta$ and α is an R-constraint.
Question: Does there exist $\Phi \subseteq \Delta$ such that

1. $\psi \in \Phi$ and
2. (Φ, α) is an argument?

Also, it is not clear how to get rid of the equality constraints. Indeed identifying variables that are connected by equality constraints does not necessarily preserve minimality of the support.

For these two reasons, it is not clear how to prove that the complexity of the argumentation problems only depends on the relational clone $\langle\Gamma\rangle$. The best we can obtain is the following key lemma, which will be of use for the classifications for Arg-Check and Arg-Rel.

Lemma 4.2. *Let Γ, Γ' be two constraint languages and R a Boolean relation. If $\Gamma' \subseteq \langle\Gamma\rangle_{\neq}$ and $R \in \langle\Gamma\rangle_{\nparallel,\neq}$ then*

1. Arg-Check$(\Gamma', R) \leq_{\mathrm{m}}^{\log}$ Arg-Check(Γ).
2. Arg-Rel$(\Gamma', R) \leq_{\mathrm{m}}^{\log}$ Arg-Rel(Γ).

Proof. 1. Let (Φ, α) be an instance of the first problem, where $\Phi = \{\delta_i \mid i \in I\}$ and $\alpha = R(x_1, \ldots, x_k)$. We map this instance to (Φ', α'), where $\Phi' = \{\delta_i' \mid \delta_i \in \Phi\}$ and α' is a Γ-formula equivalent to $R(x_1, \ldots, x_k)$. For $i \in I$ we obtain δ_i' from δ_i by replacing δ_i by an equivalent Γ-formula with existential quantifiers (such a representation exists since $\Gamma' \subseteq \langle\Gamma\rangle_{\neq}$) and deleting all existential quantifiers.

2. Let $(\Delta, \alpha, \delta_1)$ be an instance of the first problem, where $\Delta = \{\delta_i \mid i \in I\}$ and $\alpha = R(x_1, \ldots, x_k)$. We map this instance to $(\Delta', \alpha', \delta_1')$, where $\Delta' = \{\delta_i' \mid \delta_i \in \Delta\}$ and α' is a Γ-formula equivalent to $R(x_1, \ldots, x_k)$. For $i \in I$ we obtain δ_i' from δ_i by the same procedure as in the previous case.

□

As we discussed above the complexity of the verification and the relevance problem when restricted to Γ-formulæ is not *a priori* completely determined by the relational clone $\langle\Gamma\rangle$. However due to the above lemma, the lattice of Boolean co-clones together with the mentioned Galois connection will still be of help.

4.2 Some co-clones and various expressibility lemmas

In this subsection we recall the relevant knowledge on the lattice of co-clones and give some technical expressibility results that will be of use for the proofs.

For the results referring to the lattice of co-clones we use the notations and the results from [CKZ08].

Lemma 4.3. *The smallest co-clone that contains all* positive *(resp.,* negative*) relations is* IS_0 *(resp.,* IS_1*). A relation* R *is in* IS_0 *(resp.,* IS_1*) if and only if* $m, m' \in R$ *implies* $m \to m' \in R$ *(resp.,* $m \not\to m' \in R$*), where the binary operator* $\to$ *(resp.,* $\not\to$*) applied on Boolean vectors is applied coordinate-wise.*

Remark 4.4. *Observe that there are relations in* IS_0 *(resp.,* IS_1*) which are not* positive *(resp.,* negative*), for instance the equality relation.*

Lemma 4.5. *Let* Γ *be a constraint language which is not Schaefer.*

- *(*II*) If* Γ *is not* complementive*,* 0-valid*,* 1-valid*, then* $\langle\Gamma\rangle$ *contains all relations that are both* 0-valid *and* 1-valid*,* $\langle\Gamma\rangle = \mathsf{II}$*.*
- *(*II_1*/*II_0*) If* Γ *is not* complementive*, not* 0-valid *but* 1-valid *(resp. not* 1-valid *but* 0-valid*), then* $\langle\Gamma\rangle$ *contains all relations that are* 1-valid *(*0-valid*),* $\langle\Gamma\rangle = \mathsf{II}_1$ *(*$\langle\Gamma\rangle = \mathsf{II}_0$*).*
- *(*II_2*) If* Γ *is not* complementive*, not* 0-valid*, not* 1-valid*, then* $\langle\Gamma\rangle$ *contains all relations,* $\langle\Gamma\rangle = \mathsf{II}_2$*.*

Let us now give some expressibility results that we will use in our hardness proofs. In the proofs of the following lemmas $V = \{x_1, \ldots, x_k\}$ will denote a set of k distinct variables. We will suppose w.l.o.g that the constraint language Γ consists of a single relation R of arity k. The reason why we can assume this is that, w.r.t. expressivity, any finite $\Gamma = \{R_1, \ldots, R_n\}$ can be 'condensed' to a single relation by the Cartesian product $R = R_1 \times \ldots \times R_n$. It clearly holds that $R \in \langle\Gamma\rangle_{\not\exists,\neq}$ and R has all properties that Γ has.

Lemma 4.6. *Let* Γ *be a constraint language. If* Γ *is*

1. complementive*, but neither* 1-valid *nor* 0-valid*, then* $(x \neq y) \in \langle\Gamma\rangle_{\not\exists,\neq}$*;*
2. *not* complementive*, but* 1-valid *and* 0-valid*, then* $(x \to y) \in \langle\Gamma\rangle_{\not\exists,\neq}$*;*
3. *neither* complementive *nor* 1-valid*, nor* 0-valid*, then* $(x \wedge \neg y) \in \langle\Gamma\rangle_{\not\exists,\neq}$*;*
4. 1-valid *and not* 0-valid*, then* $\mathrm{T} \in \langle\Gamma\rangle_{\not\exists,\neq}$*.*

Proof. Folklore, see e.g., [CKS01]. □

Lemma 4.7. *Let* Γ *be a constraint language that is both* 0-valid *and* 1-valid*. Then* $(x = y) \in \langle\Gamma\rangle_{\not\exists,\neq}$*.*

Proof. Let $R \in \Gamma$ a k-ary relation. Since $R \neq \{0,1\}^k$ there is an $m \notin R$ and $m \neq 0^k$ and $m \neq 1^k$. For $i \in \{0,1\}$, set $V_i = \{x \mid x \in V, m(x) = i\}$. We observe that the sets V_0 and V_1 are nonempty (since $m \neq 0^k$ and $m \neq 1^k$). Denote by C the $\{R\}$-constraint $C = R(x_1, \ldots, x_k)$. Set $M(x,y) = C[V_0/x, V_1/y]$. It contains $\{00, 11\}$ (since R is 0-valid and 1-valid) but not 01 (since $m \notin R$). Finally, we have $M(x,y) \wedge M(y,x) \equiv (x = y)$. □

Lemma 4.8. *Let Γ be a constraint language. If Γ is:*

1. 1-valid *but neither* 0-valid *nor* positive, *then* $(x = y) \wedge z \in \langle\Gamma\rangle_{\not\equiv,\neq}$;
2. 0-valid *but neither* 1-valid *nor* negative, *then* $(x = y) \wedge \neg z \in \langle\Gamma\rangle_{\not\equiv,\neq}$.

Proof. We only prove the first case, the second case can be treated analogously / dually. Let w.l.o.g. $\Gamma = \{R\}$, thus R is 1-valid but neither 0-valid nor positive. We perform a case distinction according to whether $R \in \mathsf{IS}_0$ or not.

Let us first suppose that $R \in \mathsf{IS}_0$. According to [CKZ08] the relation R ($\in \mathsf{IS}_0$) can be written as a conjunction of positive clauses and equalities. If R can be written with no equality, then R is positive, a contradiction. So, any representation of R as a conjunction of positive clauses and equalities requires at least one equality. Suppose thus w.l.o.g that $R(x_1, \ldots, x_k) \models (x_1 = x_2)$, while $R(x_1, \ldots, x_k) \not\models x_1$ (which means that the equality $x_1 = x_2$ can be transitively deduced from the equality constraints occurring in any representation of R). Let $W := \{x_i \mid R(x_1, \ldots, x_k) \models (x_1 = x_i)\}$. Observe that $W' = V \setminus (W \cup \{x_1\})$ is nonempty for R is not 0-valid. Denote by C the $\{R\}$-constraint $C = R(x_1, \ldots, x_k)$. Consider the constraint $M(x_1, x_2, y) = C[W/x_2, W'/y]$. One verifies that $M(x_1, x_2, y) \equiv (x_1 = x_2) \wedge y$. Therefore, $(x = y) \wedge z \in \langle\Gamma\rangle_{\not\equiv,\neq}$.

Let us now suppose that $R \notin \mathsf{IS}_0$. According to Lemma 4.3 there are $m_1, m_2 \in R$ such that $m_1 \to m_2 \notin R$. For $i, j \in \{0, 1\}$, set $V_{i,j} = \{x \mid x \in V, m_1(x) = i \wedge m_2(x) = j\}$. Observe that the sets $V_{0,0}$ and $V_{1,0}$ are nonempty (otherwise $m_2 = m_1 \to m_2$ or $1^k = m_1 \to m_2$, a contradiction). Denote by C the $\{R\}$-constraint $C = R(x_1, \ldots, x_k)$. Set $M(x_1, x_2, x_3, x_4) = C[V_{0,0}/x_1, V_{1,0}/x_2, V_{0,1}/x_3, V_{1,1}/x_4]$. It contains $\{1111, 0101, 0011\}$ (since resp. R is 1-valid, $m_1 \in R$, $m_2 \in R$) but not 1011 (since $m_1 \to m_2 \notin R$). We conclude that $M(x, y, z, z) \wedge \mathrm{T}(z)$ contains $\{111, 001\}$ but not 101. Finally, we verify that $M(x, y, z, z) \wedge M(y, x, z, z) \wedge \mathrm{T}(z) \equiv (x = y) \wedge z$. Since by Lemma 4.6, $\mathrm{T} \in \langle\Gamma\rangle_{\not\equiv,\neq}$, we obtain $(x = y) \wedge z \in \langle\Gamma\rangle_{\not\equiv,\neq}$. □

Lemma 4.9. *Let Γ be a constraint language. If Γ is not* Schaefer, *then $(x = y) \in \langle\Gamma\rangle_{\neq}$. In particular, for any relation R, if $R \in \langle\Gamma\rangle$ and Γ is not* Schaefer, *then $R \in \langle\Gamma\rangle_{\neq}$.*

Proof. We perform a case distinction according to whether Γ is 0/1-valid or not.

If Γ is 0-valid and 1-valid, then according to Lemma 4.7 there is a Γ-formula equivalent to $(x = y)$.

If Γ is not 0-valid but 1-valid (resp. not 1-valid but 0-valid), then according to Lemma 4.8 there is a Γ-formula $\varphi(x, y, z)$ equivalent to $(x = y) \wedge z$ (resp. $(x = y) \wedge \neg z$). Hence, $\exists z\, \varphi(x, y, z)$ fulfills our needs.

At last let Γ be neither 0-valid nor 1-valid. It suffices here to show that we are able to express disequality, $(x \neq y)$, since $(x = y) \equiv \exists z (x \neq z) \wedge (z \neq y)$. If Γ is complementive we conclude by Lemma 4.6, first item. Therefore suppose now that Γ is not complementive. Let w.l.o.g. $\Gamma = \{R\}$. Since R is not Horn, there are $m_1, m_2 \in R$ such that $m_1 \wedge m_2 \notin R$. For $i, j \in \{0, 1\}$, set $V_{i,j} = \{x \mid x \in V, m_1(x) = i \wedge m_2(x) = j\}$. Observe that the sets $V_{0,1}$ and $V_{1,0}$ are nonempty (otherwise $m_1 = m_1 \wedge m_2$ or $m_2 = m_1 \wedge m_2$, a contradiction). Denote by C the $\{R\}$-constraint $C = R(x_1, \ldots, x_k)$. Set $M_1(u, x, y, v) = C[V_{0,0}/u, V_{0,1}/x, V_{1,0}/y, V_{1,1}/v]$. It contains $\{0011, 0101\}$ (since $m_1, m_2 \in R$) but it does not contain 0001 (since $m_1 \vee m_2 \notin R$). Further, since R is not dualHorn, there are $m_3, m_4 \in R$ such that $m_3 \vee m_4 \notin R$. For $i, j \in \{0, 1\}$, set $V'_{i,j} = \{x \mid x \in V,\ m_3(x) = i \wedge m_4(x) = j\}$. Observe that the sets $V'_{0,1}$ and $V'_{1,0}$ are nonempty. Set $M_2(u, x, y, v) = C[V'_{0,0}/u, V'_{0,1}/x, V'_{1,0}/y, V'_{1,1}/v]$. It contains $\{0011, 0101\}$

(since $m_3, m_4 \in R$) but it does not contain 0111 (since $m_3 \vee m_4 \notin R$). Finally consider the $\{R, (t \wedge \neg f)\}$-formula

$$M(x, y, f, t) = M_1(f, x, y, t) \wedge M_2(f, x, y, t) \wedge (t \wedge \neg f).$$

One verifies that it is equivalent to $(x \neq y) \wedge (t \wedge \neg f)$. Due to the third item of Lemma 4.6, $(t \wedge \neg f)$ is expressible as a Γ-formula, and therefore so is $M(x, y, f, t)$. We conclude observing that $\exists t, f\ M(x, y, f, t)$ is equivalent to $(x \neq y)$. □

5 The complexity of Arg

The complexity of deciding the existence of an argument rests on two sources: finding a candidate support, and checking that it is consistent and proves α. Observe that the minimality condition plays no role here: there exists a minimal support if and only if there exists a support. Therefore the problem $\textsc{Arg}(\Gamma)$ lies in the class Σ_2^{p}. When there is a natural candidate as a support, then the complexity of $\textsc{Arg}(\Gamma)$ drops to the class coNP, whereas when satisfiability and implication are tractable then the complexity drops to the class NP.

Proposition 5.1. *Let* Γ *be a constraint language which is* Schaefer*, but neither* 1-valid*, nor* 0-valid*. Then* $\textsc{Arg}(\Gamma)$ *is* NP*-complete.*

Proof. The NP-membership follows from the fact that since Γ is Schaefer $\textsc{Sat}(\Gamma)$ and $\textsc{Imp}(\Gamma)$ are in P and thus a guessed argument can be verified in polynomial time. For the hardness proof we perform a case distinction according to whether Γ is complementive or not. Suppose first that every relation in Γ is complementive. We prove the following sequence of reductions:

$$\text{3-}\textsc{Sat} \leq_{\mathrm{m}}^{\log} \textsc{Arg}(\{x \neq y\}) \leq_{\mathrm{m}}^{\log} \textsc{Arg}(\Gamma).$$

The last reduction holds by Item 1 in Lemma 4.6. For the first reduction let $\varphi = \bigwedge_{i=1}^{k} C_i$ be an instance of 3-SAT where $\mathrm{var}(\varphi) = \{x_1, \ldots, x_n\}$. Let $c_1, \ldots, c_k$, $x'_1, \ldots, x'_n$, f be fresh variables. We map φ to (Δ, α) where

$$\begin{aligned}
\Delta = \quad & \textstyle\bigcup_{j=1}^{n} \{x_j \neq f, x'_j \neq f\} \\
\cup \quad & \{\textstyle\bigwedge_{j=1}^{n} (x_j \neq x'_j)\} \\
\cup \quad & \textstyle\bigcup_{i=1,\ldots,k, j=1,\ldots,n} \{x_j \neq c_i \mid \neg x_j \in C_i\} \cup \{x'_j \neq c_i \mid x_j \in C_i\}, \\
\alpha = \quad & \textstyle\bigwedge_{i=1}^{k} (c_i \neq f) \wedge \bigwedge_{j=1}^{n} (x_j \neq x'_j).
\end{aligned}$$

One can check that φ is satisfiable if and only if there exists a $\Phi \subseteq \Delta$ such that (Φ, α) is an argument. Intuitively, x'_j plays the role of $\neg x_j$, for every j at most one of the constraints $x_j \neq f$ and $x'_j \neq f$ can appear in the support of an argument, thus allowing to identify true literals, while for every i the constraints $x_j \neq c_i$ and $x'_j \neq c_i$ are used to certify that the clause C_i is satisfied.

Second, let us suppose that Γ is not complementive. We prove the following:

$$\textsc{Pos-1-in-3-Sat} \leq_{\mathrm{m}}^{\log} \textsc{Arg}(\{x \wedge \neg y\}) \leq_{\mathrm{m}}^{\log} \textsc{Arg}(\Gamma).$$

The last reduction follows by Item 3 in Lemma 4.6. For the first one we start from the NP-complete problem Pos-1-in-3-Sat in which the instance is a set of positive 3-clauses and the question is to decide whether there exists a truth assignment such that each clause contains exactly one true variable. Let $\varphi = \bigwedge_{i=1}^{k}(x_i \vee y_i \vee z_i)$ be an instance of the first problem and let $c_1, \ldots, c_k$, f be fresh variables. We map φ to (Δ, α) where

$$\begin{aligned}\Delta = \quad & \bigcup_{i=1}^{k}\{c_i \wedge x_i \wedge \neg y_i \wedge \neg z_i \wedge \neg f\} \\ \cup \quad & \bigcup_{i=1}^{k}\{c_i \wedge \neg x_i \wedge y_i \wedge \neg z_i \wedge \neg f\} \\ \cup \quad & \bigcup_{i=1}^{k}\{c_i \wedge \neg x_i \wedge \neg y_i \wedge z_i \wedge \neg f\}, \\ \alpha = \quad & (c_1 \wedge \neg f) \wedge \ldots \wedge (c_k \wedge \neg f).\end{aligned}$$

Observe that every formula in Δ can be written as a $\{x \wedge \neg y\}$-formula. One can check that there is a truth assignment such that each clause C_i contains exactly one variable set to true if and only if (Δ, α) admits an argument. Observe that for every i such an argument contains exactly one of the three formulæ involving c_i, thus providing a desired satisfying assignment. □

Proposition 5.2. *Let* Γ *be a constraint language which is neither* Schaefer*, nor* 1-valid*, nor* 0-valid*. Then* $\textsc{Arg}(\Gamma)$ *is* Σ_2^{p}*-complete.*

Proof. We give a reduction from P-$\textsc{Abd}(\Gamma)$ which is Σ_2^{p}-complete according to [NZ08]. We perform a case distinction according to whether Γ is complementive or not.

Suppose first that every relation in Γ is complementive. We show:

$$\text{P-}\textsc{Abd}(\Gamma) \leq_{\mathrm{m}}^{\log} \textsc{Arg}(\Gamma \cup \{x \neq y\}) \leq_{\mathrm{m}}^{\log} \textsc{Arg}(\Gamma).$$

The last reduction follows by Item 1 in Lemma 4.6. For the first one we map (φ, H, q), an instance of $\textsc{Abd}(\Gamma)$, to (Δ, α), where we introduce a fresh variable f and define

$$\Delta = \{\varphi\} \cup \{(h \neq f) \mid h \in H\}, \qquad \alpha = (q \neq f).$$

The proof that the reduction is correct relies on the fact that all formulæ occurring in the so obtained instance are complementive, i.e., it suffices to observe correctness for $(\Delta[f/0], \alpha[f/0])$.

In the case where Γ is not complementive we show

$$\text{P-}\textsc{Abd}(\Gamma) \leq_{\mathrm{m}}^{\log} \textsc{Arg}(\Gamma \cup \{x \wedge \neg y\}) \leq_{\mathrm{m}}^{\log} \textsc{Arg}(\Gamma).$$

The last reduction follows by Item 3 in Lemma Lemma 4.6. For the first one we map (φ, H, q), an instance of the first problem, to (Δ, α), where we introduce two fresh variables t, f and define

$$\Delta = \{\varphi\} \cup \{h \wedge \neg f \mid h \in H\} \cup \{t \wedge \neg f\}, \qquad \alpha = (q \wedge \neg f) \wedge (t \wedge \neg f).$$

Observe that Δ is made of Γ- and $\{x \wedge \neg y\}$-formulæ. It is easy to check that (φ, H, q) is a positive instance of the abduction problem if and only if there exists a support for α in Δ. □

We are now in a position to state the classification theorem.

Theorem 5.3. *Let* Γ *be a constraint language. The decision problem* $\textsc{Arg}(\Gamma)$ *is*

1. *in* P *if* Γ *is* Schaefer *and* ε*-valid,*
2. NP*-complete if* Γ *is* Schaefer *and not* ε*-valid,*
3. coNP*-complete if* Γ *is not* Schaefer *and* ε*-valid,*
4. Σ_2^{p}*-complete if* Γ *is not* Schaefer *and not* ε*-valid.*

Proof. 1. One easily observes that, due to the fact that Γ is 1-valid or 0-valid, an instance (Δ, α) of $\textsc{Arg}(\Gamma)$ has a solution if and only if Δ implies α. This condition can be checked in polynomial time since Γ is Schaefer and thus $\textsc{Imp}(\Gamma)$ is in P.

2. Follows from Proposition 5.1.

3. One easily observes that, due to the fact that Γ is 1-valid or 0-valid, an instance (Δ, α) of $\textsc{Arg}(\Gamma)$ has a solution if and only if Δ implies α. This condition can be checked in coNP since $\textsc{Imp}(\Gamma)$ is in coNP.

   To prove coNP-hardness we give a reduction from the coNP-complete problem $\textsc{Imp}(\Gamma)$. We map (φ, ψ) an instance of the first problem to $(\{\varphi\}, \psi)$.

4. Follows from Proposition 5.2.

□

6 The complexity of Arg-Check

In this section we give the complexity classification for the verification problem. As discussed in Section 4.1 the Galois connection does not hold *a priori* for this problem. However, the following theorem shows that it holds *a posteriori*, this means that the dichotomy follows the borders of Post's lattice, i.e., the complexity of $\textsc{Arg-Check}(\Gamma)$ depends on the relational clone $\langle\Gamma\rangle$ only.

Theorem 6.1. *Let* Γ *be a constraint language. Then the decision problem* $\textsc{Arg-Check}(\Gamma)$ *is*

1. *in* P *if* Γ *is* Schaefer*,*
2. DP*-complete if* Γ *is not* Schaefer*.*

The argument verification problem is in DP. Indeed $\textsc{Arg-Check} = A \cap B$, with $A = \{(\Delta, \Phi, \alpha) \mid \Phi \text{ is satisfiable}, \forall \varphi \in \Phi : \Phi \setminus \{\varphi\} \not\models \alpha\}$ and $B = \{(\Delta, \Phi, \alpha) \mid \Phi \models \alpha\}$, and $A \in \mathrm{NP}$ and $B \in \mathrm{coNP}$.

Proposition 6.2. *Let* Γ *be a constraint language that is* Schaefer*. Then* $\textsc{Arg-Check}(\Gamma)$ *is in* P*.*

Proof. Use that $\textsc{Sat}(\Gamma)$ and $\textsc{Imp}(\Gamma)$ are in P. □

Proposition 6.3. *Let* Γ *be a constraint language which is neither* Schaefer *nor* complementive*. Then* $\textsc{Arg-Check}(\Gamma)$ *is* DP*-complete.*

Proof. For the hardness we give a reduction from Critical-Sat, a DP-complete problem according to [PW88]. We will use as an intermediate problem the variant of Arg-Check with two parameters, Arg-Check(Γ', R) as defined in Section 4.1, differentiating the restrictions put on the knowledge base from the ones put on the claim.

We perform a case distinction according to whether Γ is 0-valid and/or 1-valid. Throughout the proof we denote by $\varphi = \bigwedge_{i=1}^{k} C_i$ an instance of Critical-Sat.

Suppose first that Γ is both 0-valid and 1-valid. We prove for some well-chosen constraint language $\Gamma' \subseteq \langle\Gamma\rangle$ the following sequence of reductions:

$$\begin{aligned} \text{Critical-Sat} &\leq_{\mathrm{m}}^{\log} \text{Arg-Check}(\Gamma', x \to y) \\ &\leq_{\mathrm{m}}^{\log} \text{Arg-Check}(\Gamma). \end{aligned}$$

For the first reduction we associate with φ the instance (Φ, α) where

$$\begin{aligned} \Phi &= \{C_i \vee (f \to t) \mid i = 1, \ldots, k\}, \\ \alpha &= (f \to t), \end{aligned}$$

with f, t fresh variables. It is easy to see that φ is a critical instance if and only if (Φ, α) is an argument.

For the second reduction observe that all formulæ in Φ are constraints built upon a finite set Γ' of relations which are 1-valid and 0-valid and thus $\Gamma' \subseteq \langle\Gamma\rangle$ according to Lemma 4.5. Since Γ is not Schaefer, following Lemma 4.9 we have $\Gamma' \subseteq \langle\Gamma\rangle_{\neq}$. Further, the relation $x \to y$ can be expressed by a Γ-formula according to Item 2 of Lemma 4.6. With this, the second reduction follows by Lemma 4.2.

Suppose now that Γ is 1-valid and not 0-valid. The other case (0-valid and not 1-valid) can be treated analogously. We show for some well-chosen constraint language $\Gamma' \subseteq \langle\Gamma\rangle$ that

$$\begin{aligned} \text{Critical-Sat} &\leq_{\mathrm{m}}^{\log} \text{Arg-Check}(\Gamma', \mathrm{T}) \\ &\leq_{\mathrm{m}}^{\log} \text{Arg-Check}(\Gamma). \end{aligned}$$

For the first reduction we associate with φ the instance (Φ, α) where

$$\begin{aligned} \Phi &= \{C_i \vee u \mid i = 1, \ldots, k\}, \\ \alpha &= u, \end{aligned}$$

with u being a fresh variable. It is easy to see that φ is a critical instance if and only if (Φ, α) is an argument.

For the second reduction observe that all formulæ in Φ are constraints built upon a finite set Γ' of relations which are 1-valid and thus $\Gamma' \subseteq \langle\Gamma\rangle$ according to Lemma 4.5. Since Γ is not Schaefer, following Lemma 4.9 we have $\Gamma' \subseteq \langle\Gamma\rangle_{\neq}$. Further, the relation T can be expressed by a Γ-formula according to Lemma 4.6. With this, the second reduction follows by Lemma 4.2.

Finally suppose that Γ is neither 1-valid nor 0-valid. We show for some well-chosen constraint language $\Gamma' \subseteq \langle\Gamma\rangle$ that

$$\begin{aligned} \text{Critical-Sat} &\leq_{\mathrm{m}}^{\log} \text{Arg-Check}(\Gamma', x \wedge \neg y) \\ &\leq_{\mathrm{m}}^{\log} \text{Arg-Check}(\Gamma). \end{aligned}$$

For the first reduction we associate with φ the instance (Φ, α) where

$$\begin{aligned}\Phi &= \{(C_i \vee u) \wedge \neg v \mid i = 1, \ldots, k\},\\ \alpha &= u \wedge \neg v,\end{aligned}$$

with u, v fresh variables. It is easy to see that φ is a critical instance if and only if (Φ, α) is an argument.

For the second reduction observe that all formulæ in Φ are constraints built upon a finite set Γ' of relations and thus $\Gamma' \subseteq \langle\Gamma\rangle$ according to Lemma 4.5. Since Γ is not Schaefer, following Lemma 4.9 we have $\Gamma' \subseteq \langle\Gamma\rangle_{\neq}$. Further, the relation $x \wedge \neg y$ can be expressed by a Γ-formula according to Item 3 of Lemma 4.6. With this, the second reduction follows by Lemma 4.2. □

Proposition 6.4. *Let* Γ *be a constraint language which is not* Schaefer *but is* complementive. *Then* Arg-Check(Γ) *is* DP*-complete.*

Proof. We prove that Arg-Check($\Gamma \cup \{\mathrm{T}\}$) $\leq_{\mathrm{m}}^{\log}$ Arg-Check(Γ). This will prove hardness for Arg-Check(Γ) since $\Gamma \cup \{T\}$ is neither Schaefer nor complementive (because of T) and therefore Arg-Check($\Gamma \cup \{T\}$) is a DP-complete problem according to Proposition 6.3.

Suppose first that Γ is both 0-valid and 1-valid. Then we show that Arg-Check($\Gamma \cup \{\mathrm{T}\}$) $\leq_{\mathrm{m}}^{\log}$ Arg-Check($\Gamma \cup \{=\}$) $\leq_{\mathrm{m}}^{\log}$ Arg-Check(Γ). The second reduction holds according to Lemma 4.7. For the first one let (Φ, α) be an instance of Arg-Check($\Gamma \cup \{\mathrm{T}\}$). Introduce a fresh variable t and replace in all formulæ all T-constraints $\mathrm{T}(x)$ by $(x = t)$. Thus we obtain (Φ', α') an instance of Arg-Check($\Gamma \cup \{=\}$). The key to observe that this reduction is correct is that Γ is complementive, i.e., it suffices to observe correctness for $(\Phi'[t/1], \alpha[t/1])$.

In the case Γ is neither 0-valid nor 1-valid, then we show that Arg-Check($\Gamma \cup \{\mathrm{T}\}$) $\leq_{\mathrm{m}}^{\log}$ Arg-Check($\Gamma \cup \{\neq\}$) $\leq_{\mathrm{m}}^{\log}$ Arg-Check(Γ). The second reduction holds according to Item 1 of Lemma 4.6. For the first one let (Φ, α) be an instance of Arg-Check($\Gamma \cup \{\mathrm{T}\}$). Introduce a fresh variable t and introduce in all formulæ for each T-constraint $\mathrm{T}(x)$ a new variable f_x and replace $\mathrm{T}(x)$ by the two disequality constraints $(x \neq f_x)$ and $(f_x \neq t)$. Thus we obtain (Φ', α') an instance of Arg-Check($\Gamma \cup \{\neq\}$). Again, the key to observe correctness is that one may restrict attention to the case $t = 1$. □

7 The complexity of Arg-Rel

In this section we give the complexity classification for the relevance problem. As for the verification problem the Galois connection does not hold *a priori* for this problem (see the discussion in Section 4.1). However, interestingly and contrary to the verification problem, it does not hold *a posteriori* either (unless P = NP). We reveal constraint languages Γ_1 and Γ_2 such that $\langle\Gamma_1\rangle = \langle\Gamma_2\rangle$, and Arg-Rel($\Gamma_1$) is in P whereas Arg-Rel(Γ_2) is NP-complete. As we will see, it is the equality relation, =, that is responsible for the discrepancy in complexity.

Theorem 7.1. *Let* Γ *be a constraint language. Then the decision problem* Arg-Rel(Γ) *is*

1. *in* P *if* Γ *is positive or negative,*
2. NP*-complete if* Γ *is* Schaefer *but neither positive, nor negative,*

3. Σ_2^{p}*-complete if* Γ *is not* Schaefer.

Remark 7.2. *Observe that we inherit all hardness-results from* Arg(Γ) *via the reduction* $(\Delta, \alpha) \mapsto (\Delta \cup \{\phi\}, \alpha \wedge \phi, \phi)$*, where* ϕ *is any non-trivial* Γ*-formula made of fresh variables.*

While the polynomial cases of Arg and Arg-Check relied directly on the tractability of Sat and Imp, for Arg-Rel we need to investigate the structure of the problem.

Proposition 7.3. *Let* Γ *be a constraint language. If* Γ *is either* positive *or* negative*, then* Arg-Rel(Γ) *is in* P*.*

Proof. We treat only the case of positive Γ, the other case can be treated analogously / dually. In this case α and the formulæ in the knowledge base Δ can be considered as positive CNF-formulæ. We claim that Algorithm 1 decides Arg-Rel(Γ) in polynomial time. The running

Algorithm 1 Algorithm for Arg-Rel(Γ) for positive Γ.

Require: a set Δ of positive formulæ, $\psi \in \Delta$ and a positive formula $\alpha = \bigwedge_{i \in I} C_i$.
  for all $i \in I$ **do**
    $\Delta_i := \{\psi\} \cup \{\delta \in \Delta \mid \delta \not\models C_i\}$
    if $\Delta_i \models \alpha$ **then**
      accept
    end if
  end for
  reject

time of Algorithm 1 is obviously polynomial (the test $\Delta_i \models \alpha$ is an instance of Imp(Γ), which is in P for positive Γ).

To prove correctness, we need the following easy but crucial observation.

Observation 7.4. *Let* a, b *be positive CNF-formulæ and let* γ *be a positive clause. If* $a \not\models \gamma$ *and* $b \not\models \gamma$*, then* $a \wedge b \not\models \gamma$*.*

If Algorithm 1 accepts, then there exists a $\Delta_i \subseteq \Delta$ such that $\Delta_i \models \alpha$ and no $\delta \in \Delta_i \setminus \{\psi\}$ entails C_i. With Observation 7.4 we obtain that $\Delta_i \setminus \{\psi\} \not\models C_i$, therefore $\Delta_i \setminus \{\psi\} \not\models \alpha$. We conclude that Δ_i contains a minimal support Φ such that $\psi \in \Phi$.

Conversely, let Φ be a minimal support such that $\psi \in \Phi$. Since $\Phi \setminus \{\psi\} \not\models \alpha$, there is at least one i such that $\Phi \setminus \{\psi\} \not\models C_i$, i.e., in particular no $\delta \in \Phi \setminus \{\psi\}$ entails C_i. For this i the algorithm constructs $\Delta_i := \{\psi\} \cup \{\delta \in \Delta \mid \delta \not\models C_i\}$. Obviously $\Phi \subseteq \Delta_i$, and since $\Phi \models \alpha$ we obtain that $\Delta_i \models \alpha$ which causes the algorithm to accept.

Note that the same algorithmic idea was applied in [CSTW11, Proposition 3.8] to solve the relevance problem for positive terms. □

Let us now turn to the NP-complete case, when Γ is Schaefer but neither positive nor negative. Observe that if Γ is Schaefer, then Arg-Check(Γ) is in P (see Theorem 6.1), and therefore Arg-Rel(Γ) is in NP: Guess a Φ and verify that $\psi \in \Phi$ and $(\Phi, \alpha) \in$ Arg-Check. The hardness proofs rely on the following basic hardness results.

Lemma 7.5. Arg-Rel($\{x = y\}$)*,* Arg-Rel($\{(x = y) \wedge z\}$) *and* Arg-Rel($\{(x = y) \wedge \neg z\}$) *are* NP*-hard.*

Proof. For Arg-Rel($\{x = y\}$) we give a reduction from 3-Sat. Let $\varphi = \bigwedge_{i=1}^{k} C_i$, $\mathrm{var}(\varphi) = \{x_1, \ldots, x_n\}$. Let $c_0, c_1, \ldots c_k, s$ be fresh variables. We map φ to the instance (Δ, α, ψ) defined as follows.

$$\begin{aligned}
\Delta = &\quad \{\gamma_j, \delta_j \mid 1 \leq j \leq n\} \cup \{\psi\} \\
\gamma_j = &\quad (c_0 = x_j) \wedge \textstyle\bigwedge_{i \text{ s.t. } x_j \in C_i} (c_{i-1} = c_i) \\
\delta_j = &\quad (x_j = s) \wedge \textstyle\bigwedge_{i \text{ s.t. } \neg x_j \in C_i} (c_{i-1} = c_i) \\
\alpha = &\quad (c_0 = s) \\
\psi = &\quad (c_k = s)
\end{aligned}$$

Correctness is not difficult to observe. There is a one-to-one correspondence between (not necessarily minimal) supports Φ in which ψ is relevant and satisfying assignments σ for φ given by $\gamma_j \in \Phi$ iff $\sigma(x_j) = 1$ and $\delta_j \in \Phi$ iff $\sigma(x_j) = 0$. A support (containing a relevant ψ) does never contain both γ_j and δ_j, since otherwise ψ would not be relevant.

For Arg-Rel($\{(x = y) \wedge z\}$) (resp. Arg-Rel($\{(x = y) \wedge \neg z\}$)) we use the same reduction scheme as above, we introduce a new variable t and replace any equality of the form $(x = y)$ by $(x = y) \wedge t$ (resp. $(x = y) \wedge \neg t$). □

Proposition 7.6. *Let* Γ *be a constraint language. If* Γ *is* Schaefer *but neither* positive *nor* negative *then* Arg-Rel(Γ) *is* NP*-complete.*

Proof. It remains to show NP-hardness. If Γ is 0-valid and 1-valid, we conclude with Lemma 4.7 and Lemma 7.5. If Γ is not 0-valid but 1-valid (resp. 0-valid but not 1-valid), we conclude with Lemma 4.8 and Lemma 7.5. If Γ is neither 0-valid nor 1-valid, Arg(Γ) is NP-hard and we conclude with Remark 7.2. □

To conclude the proof of Theorem 7.1 it remains to deal with the Σ_2^{p}-complete cases.

Proposition 7.7. *Let* Γ *be not* Schaefer*. Then* Arg-Rel(Γ) *is* Σ_2^{p}*-complete.*

Proof. Membership in Σ_2^{p} follows as for Arg: given an instance (Δ, α, ψ), guess a support $\Phi \subseteq \Delta$, verify that $\psi \in \Phi$, and subsequently check with an NP-oracle that Φ is consistent, Φ entails α and that Φ is minimal w.r.t. the last property.

We turn to the hardness proof. We will use the problem Arg-Rel(Γ', R) with two parameters, in which we differentiate the restrictions put on the knowledge base and the claim, as an intermediate problem.

If Γ is complementive, we can apply the same trick as in Proposition 6.4 in order to reduce from the non-complementive case. It therefore suffices to show hardness for non-complementive Γ. We perform a case distinction according to whether Γ is 0/1-valid or not.

Γ *is neither 1-valid, nor 0-valid.* For those Γ the problem Arg(Γ) is Σ_2^{p}-hard. We conclude with Remark 7.2.

Γ *is both 1-valid and 0-valid.* We give a reduction from the Σ_2^{p}-hard problem Abd(Γ). An instance is given by (φ, H, q), where φ is a Γ-formula, $H \subseteq \mathrm{var}(\varphi)$ and q is a variable. The instance (φ, H, q) is a positive one if and only if there is an $E \subseteq \mathrm{Lits}(H)$ such that $\varphi \wedge E$ is satisfiable and $\varphi \wedge E \models q$.

We give the following sequence of reductions.

$$\begin{aligned}\textsc{Abd}(\Gamma) &\leq_{\mathrm{m}}^{\log} \textsc{Arg-Rel}(\Gamma_1, x \vee \neg y \vee z)\\ &\leq_{\mathrm{m}}^{\log} \textsc{Arg-Rel}(\Gamma_2, (x = y) \wedge (z = w))\\ &\leq_{\mathrm{m}}^{\log} \textsc{Arg-Rel}(\Gamma),\end{aligned}$$

where

$$\begin{aligned}\Gamma_1 &= \Gamma \cup \{(x \vee \neg y), (x = y)\},\\ \Gamma_2 &= \Gamma_1 \cup \{R_\delta\},\\ R_\delta(x_1, \ldots, x_7) &= \big((x_1 \vee \neg x_2 \vee x_4) \leftrightarrow (x_4 = x_5)\big) \wedge (x_6 = x_7).\end{aligned}$$

We will first treat the third and the second reduction which are short and very technical. Observe that Γ_2 is both 0-valid and 1-valid. We have therefore by Lemma 4.5, first item, that $\Gamma_2 \subseteq \langle\Gamma\rangle$. Since Γ is not Schaefer, we have by Lemma 4.9 that $\Gamma_2 \subseteq \langle\Gamma\rangle_{\neq}$. Further, we have by Lemma 4.7 that $(x = y) \wedge (z = w) \in \langle\Gamma\rangle_{\not\equiv,\neq}$. Therefore, the third reduction follows by the second item of Lemma 4.2.

For the second reduction let (Δ, α, ψ) be an instance of $\textsc{Arg-Rel}(\Gamma_1, x \vee \neg y \vee z)$, where $\alpha = (x_\alpha \vee \neg y_\alpha \vee z_\alpha)$. We construct the instance $(\Delta', \alpha', \psi')$ of $\textsc{Arg-Rel}(\Gamma_2, (x = y) \wedge (z = w))$ as follows. Let u_1, u_2, v_1, v_2 be fresh variables. Then we define:

$$\begin{aligned}\Delta' &= \Delta \cup \{\delta\}\\ \alpha' &= (u_1 = u_2) \wedge (v_1 = v_2)\\ \psi' &= \psi\\ \delta &= R_\delta(x_\alpha, y_\alpha, z_\alpha, u_1, u_2, v_1, v_2)\end{aligned}$$

We observe that Δ' is a set of Γ_2-formulæ, as desired. By definition of R_δ, the formula δ is equivalent to $\big(\alpha \leftrightarrow (u_1 = u_2)\big) \wedge (v_1 = v_2)$. This allows us to observe that any support for α' will contain the formula δ which assures a one-to-one correspondence between the supports of the two instances.

It remains to give the first reduction which constitutes the main transformation idea between Abd and Arg-Rel. Let (φ, H, q) be an instance of $\textsc{Abd}(\Gamma)$, where $H = \{h_1, \ldots, h_k\}$. We construct the instance (Δ, α, ψ) of $\textsc{Arg-Rel}(\Gamma_1, x \vee \neg y \vee z)$ as follows. Let s, t, f be fresh variables. Then we define:

$$\begin{aligned}\Delta &= \{(h_i \vee \neg t), (\neg h_i \vee f) \mid 1 \leq i \leq n\} \cup \{\varphi\} \cup \{\psi\}\\ \alpha &= (s \vee \neg t \vee f)\\ \psi &= (s = q)\end{aligned}$$

We observe that Δ is a set of Γ_1-formulæ, as desired.

We now show that there is an explanation for (φ, H, q) if and only if Δ contains a minimal support for α containing ψ. For the left-to-right implication let $E \subseteq \mathrm{Lits}(H)$ such that $\varphi \wedge E$ is satisfiable and $\varphi \wedge E \models q$.

We define

$$\Phi \quad = \quad \{(h_i \vee \neg t) \mid h_i \in E\} \cup \{(\neg h_i \vee f) \mid \neg h_i \in E\} \cup \{\varphi\} \cup \{\psi\}.$$

Note that it suffices to show that

a) $\psi \in \Phi$,

b) Φ is satisfiable,

c) $\Phi \models \alpha$, and

d) $\Phi \backslash \{\psi\} \not\models \alpha$.

Such a support is not necessarily minimal, but will contain a minimal support as desired. Item a) holds by construction of Φ and item b) follows form the assumption that all formulæ are 0-valid and 1-valid.

We turn to item c). It suffices to show the following four cases.

- $\Phi[t/0, f/0] \models \alpha[t/0, f/0]$
- $\Phi[t/0, f/1] \models \alpha[t/0, f/1]$
- $\Phi[t/1, f/1] \models \alpha[t/1, f/1]$
- $\Phi[t/1, f/0] \models \alpha[t/1, f/0]$

The first three cases are obvious, since

$$\alpha[t/0, f/0] \equiv \alpha[t/0, f/1] \equiv \alpha[t/1, f/1] \equiv 1.$$

In order to show the last one, observe that $\Phi[t/1, f/0] \equiv \varphi \wedge E \wedge (s = q)$ and that $\alpha[t/1, f/0] \equiv s$. This shows that $\Phi[t/1, f/0] \models \alpha[t/1, f/0]$ since $\varphi \wedge E \models q$.

We turn to item d). It suffices to show that $(\Phi \backslash \{\psi\})[t/1, f/0] \not\models \alpha[t/1, f/0]$. But this is obvious, since $(\Phi \backslash \{\psi\})[t/1, f/0] \equiv \varphi \wedge E$ and $\alpha[t/1, f/0] \equiv s$ and $s \notin \mathrm{var}(\varphi \wedge E)$.

We now turn to the right-to-left implication. Let $\Phi \subseteq \Delta$ such that

a) $\psi \in \Phi$,

b) Φ is satisfiable,

c) $\Phi \models \alpha$, and

d) $\Phi \backslash \{\psi\} \not\models \alpha$.

We define

$$E \quad = \quad \{h_i \mid (h_i \vee \neg t) \in \Phi\} \cup \{\neg h_i \mid (\neg h_i \vee f) \in \Phi\}.$$

We first show that $\varphi \wedge E$ is satisfiable (this implies also that E is satisfiable). From d) we know that $\Phi\backslash\{\psi\} \wedge \neg\alpha$ is satisfiable. That is, $\Phi\backslash\{\psi\} \wedge \neg s \wedge t \wedge \neg f$ is satisfiable. We conclude that in particular the formula $(\Phi\backslash\{\psi\})[t/1, f/0]$ is satisfiable, which is equivalent to $\varphi \wedge E$.

From c) we conclude that in particular $\Phi[t/1, f/0] \models \alpha[t/1, f/0]$. The formula $\alpha[t/1, f/0]$ is equivalent to s. By definition of E and since $\psi \in \Phi$, the formula $\Phi[t/1, f/0]$ is either equivalent to $E \wedge \varphi \wedge \psi$ or to $E \wedge \psi$. Since $E \wedge \psi \equiv E \wedge (s = q)$ and $s, q \notin E$ we have that $E \wedge \psi \not\models s$. Thus the first case applies. That is, we have that $\varphi \wedge E \wedge (s = q) \models s$. Since $s \notin \mathrm{var}(\varphi \wedge E)$ we conclude that $\varphi \wedge E \models q$.

Γ *is 1-valid but not 0-valid (or the converse).* By duality it suffices to treat the case where Γ is 1-valid and not 0-valid. We give a reduction from the Σ_2^{p}-hard problem $\textsc{Abd}(\Gamma)$. The structure of the proof is the same as in the previous case. We give the following sequence of reductions.

$$\begin{aligned}\textsc{Abd}(\Gamma) &\leq_{\mathrm{m}}^{\log} \textsc{Arg-Rel}(\Gamma_1, x \vee y)\\ &\leq_{\mathrm{m}}^{\log} \textsc{Arg-Rel}(\Gamma_2, x \wedge y)\\ &\leq_{\mathrm{m}}^{\log} \textsc{Arg-Rel}(\Gamma),\end{aligned}$$

where

$$\begin{aligned}\Gamma_1 &= \Gamma \cup \{\mathrm{T}, (x = y), (x \vee \neg y)\},\\ \Gamma_2 &= \Gamma_1 \cup \{R_\delta\},\\ R_\delta(x_1, \ldots, x_4) &= \big((x_1 \vee x_2) \leftrightarrow x_3\big) \wedge x_4.\end{aligned}$$

Since Γ_2 is 1-valid, we have by Lemma 4.5, second item that $\Gamma_2 \subseteq \langle\Gamma\rangle$. Since Γ is not Schaefer, we have by Lemma 4.9 that $\Gamma_2 \subseteq \langle\Gamma\rangle_{\neq}$. Further, we have by Lemma 4.6, fourth item that $x \wedge y \in \langle\Gamma\rangle_{\not\equiv,\neq}$. Therefore, the third reduction follows by the second item of Lemma 4.2.

For the second reduction let (Δ, α, ψ) be an instance of $\textsc{Arg-Rel}(\Gamma_1, x \vee y)$, where $\alpha = (x_\alpha \vee y_\alpha)$. We construct the instance $(\Delta', \alpha', \psi')$ of $\textsc{Arg-Rel}(\Gamma_2, x \wedge y)$ as follows. Let u, v be fresh variables. Then we define:

$$\begin{aligned}\Delta' &= \Delta \cup \{\delta\}\\ \alpha' &= u \wedge v\\ \psi' &= \psi\\ \delta &= R_\delta(x_\alpha, y_\alpha, u, v)\end{aligned}$$

We observe that Δ' is a set of Γ_2-formulæ, as desired. By definition of R_δ, the formula δ is equivalent to $\big(\alpha \leftrightarrow u\big) \wedge v$. This allows us to observe that any support for α' will contain the formula δ which assures a one-to-one correspondence between the supports of the two instances.

For the first reduction let (φ, H, q) be an instance of $\textsc{Abd}(\Gamma)$, where $H = \{h_1, \ldots, h_k\}$. We construct the instance (Δ, α, ψ) of $\textsc{Arg-Rel}(\Gamma_1, x \vee y)$ as follows. Let s, f be fresh variables. Then we define:

$$\begin{aligned}\Delta &= \{h_i, (\neg h_i \vee f) \mid 1 \leq i \leq n\} \cup \{\varphi\} \cup \{\psi\}\\ \alpha &= s \vee f\\ \psi &= (s = q)\end{aligned}$$

Obviously, Δ is a set of Γ_1-formulæ, as desired. Correctness can be proved as in the previous case, though more easily. □

8 Conclusion and future work

In this paper we presented complete complexity classifications for three important computational tasks in argumentation, namely the existence, the validity and the relevance problem. The classifications have been obtained in Schaefer's popular framework, i.e., formulæ are in generalized conjunctive normal form and restrictions are made on the allowed type of constraints (generalized clauses). This approach covers classical classes of CNF-formulæ. For instance we obtain that the argument existence problem is NP-complete for Horn-, dualHorn-, affine- and 2CNF-formulæ, whereas the argument verification problem is tractable in these cases. Observe that the frontier between hard and easy problems for Arg-Check is the same as for the implication problem Imp. It may come as a surprise that there are fragments (for instance in the case of 0-valid-non-Schaefer relations) for which verifying an argument is potentially harder than deciding the existence of an argument (Arg-Check is DP-complete, Arg is "only" coNP-complete). Finally, the relevance problem is the hardest among the studied problems: already the equality relation makes it NP-hard. The only tractable fragment is that of positive/negative formulæ.

It would be interesting to extend the study to different variants on the claim as it has been done in [NZ08, CST11] for abduction. The complexity of the problems studied in this paper is also a computational core for evaluating more complex argumentation problems, for instance, the warranted formula problem (WFP) on argument trees, which has been shown to be PSPACE-complete [HG10]. It might be the case that fragments studied here also lower the complexity of WFP, but we leave details for future work.

Acknowledgements

The authors would like to thank Julian-Steffen Müller for the nice proof of Proposition 7.5.

References

[ABI+05] E. Allender, M. Bauland, N. Immerman, H. Schnoor, and H. Vollmer. The complexity of satisfiability problems: Refining schaefer's theorem. In *MFCS*, pages 71–82, 2005.

[BH01] P. Besnard and A. Hunter. A logic-based theory of deductive arguments. *Artif. Intell.*, 128:203–235, 2001.

[BH08] P. Besnard and A. Hunter. *Elements of Argumentation*. MIT Press, 2008.

[BKKR69] V. G. Bodnarchuk, L. A. Kalužnin, V. N. Kotov, and B. A. Romov. Galois theory for Post algebras. I, II. *Cybernetics*, 5:243–252, 531–539, 1969.

[BRSV05] Elmar Böhler, Steffen Reith, Henning Schnoor, and Heribert Vollmer. Bases for Boolean co-clones. *Inf. Process. Lett.*, 96(2):59–66, 2005.

[CKS01] N. Creignou, S. Khanna, and M. Sudan. *Complexity Classifications of Boolean Constraint Satisfaction Problems*. Monographs on Discrete Applied Mathematics. SIAM, 2001.

[CKZ08] Nadia Creignou, Phokion G. Kolaitis, and Bruno Zanuttini. Structure identification of Boolean relations and plain bases for co-clones. *J. Comput. Syst. Sci.*, 74(7):1103–1115, 2008.

[CML00] C. Chesñevar, A. Maguitman, and R. Loui. Logical models of argument. *ACM Comput. Surv.*, 32:337–383, 2000.

[Coo71] S. A. Cook. The complexity of theorem proving procedures. In *Proceedings 3rd Annual ACM Symposium on Theory of Computing*, pages 151–158. ACM Press, 1971.

[CST11] N. Creignou, J. Schmidt, and M. Thomas. Complexity classifications for propositional abduction in Post's framework. *Journal of Logic and Computation*, 2011. To appear.

[CSTW11] N. Creignou, J. Schmidt, M. Thomas, and S. Woltran. Complexity of logic-based argumentation in Post's framework. *Argument & Computation*, 2(2-3):107–129, 2011.

[CV08] N. Creignou and H. Vollmer. Boolean constraint satisfaction problems: when does Post's lattice help? In N. Creignou, Ph. G. Kolaitis, and H. Vollmer, editors, *Complexity of Constraints*, volume 5250, pages 3–37. Springer Verlag, Berlin Heidelberg, 2008.

[CZ06] N. Creignou and B. Zanuttini. A complete classification of the complexity of propositional abduction. *SIAM J. Comput.*, 36(1):207–229, 2006.

[Dun95] 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.

[EG95] T. Eiter and G. Gottlob. The complexity of logic-based abduction. *J. ACM*, 42(1):3–42, 1995.

[Gei68] D. Geiger. Closed Systems of Functions and Predicates. *Pacific Journal of Mathematics*, 27(1):95–100, 1968.

[HG10] R. Hirsch and N. Gorogiannis. The complexity of the warranted formula problem in propositional argumentation. *J. Log. Comput.*, 20:481–499, 2010.

[Jea98] P. G. Jeavons. On the algebraic structure of combinatorial problems. *Theoretical Computer Science*, 200:185–204, 1998.

[NZ08] G. Nordh and B. Zanuttini. What makes propositional abduction tractable. *Artif. Intell.*, 172(10):1245–1284, 2008.

[Pap94] C. H. Papadimitriou. *Computational Complexity*. Addison-Wesley, 1994.

[Pos41] E. Post. The two-valued iterative systems of mathematical logic. *Ann. Math. Stud.*, 5:1–122, 1941.

[PV02] H. Prakken and G. Vreeswijk. Logical systems for defeasible argumentation. In D. Gabbay, editor, *Handbook of Philosophical Logic*. Kluwer, 2002.

[PW88] C. Papadimitriou and D. Wolfe. The complexity of facets resolved. *J. Comput. Syst. Sci.*, 37(1):2–13, 1988.

[PWA03] S. Parsons, M. Wooldridge, and L. Amgoud. Properties and complexity of some formal inter-agent dialogues. *J. Log. Comput.*, 13(3):347–376, 2003.

[Sch78] T. J. Schaefer. The complexity of satisfiability problems. In *Proceedings 10th Symposium on Theory of Computing*, pages 216–226. ACM Press, 1978.

[SS08] H. Schnoor and I. Schnoor. Partial polymorphisms and constraint satisfaction problems. In N. Creignou, Ph. G. Kolaitis, and H. Vollmer, editors, *Complexity of Constraints*, volume 5250, pages 229–254. Springer Verlag, Berlin Heidelberg, 2008.