Dung's Argumentation Framework: Unveiling the Expressive Power with Inconsistent Databases

Yasir Mahmood[1], **Markus Hecher**[2,3], **Axel-Cyrille Ngonga Ngomo**[1]

[1]DICE group, Department of Computer Science, Paderborn University, Germany
[2]Univ. Artois, CNRS, UMR 8188, Centre de Recherche en Informatique de Lens (CRIL), F-62300 Lens, France
[3]CSAIL, Massachusetts Institute of Technology, United States
{yasir.mahmood, axel.ngonga}@uni-paderborn.de, hecher@mit.edu

Abstract

The connection between inconsistent databases and Dung's abstract argumentation framework has recently drawn growing interest. Specifically, an inconsistent database, involving certain types of integrity constraints such as functional and inclusion dependencies, can be viewed as an argumentation framework in Dung's setting. Nevertheless, no prior work has explored the exact expressive power of Dung's theory of argumentation when compared to inconsistent databases and integrity constraints. In this paper, we close this gap by arguing that an argumentation framework can also be viewed as an inconsistent database. We first establish a connection between subset-repairs for databases and extensions for AFs considering conflict-free, naive, admissible and preferred semantics. Further, we define a new family of attribute-based repairs based on the principle of maximal content preservation. The effectiveness of these repairs is then highlighted by connecting them to stable, semi-stable, and stage semantics. Our main contributions include translating an argumentation framework into a database together with integrity constraints. Moreover, this translation can be achieved in polynomial time, which is essential in transferring complexity results between the two formalisms.

Introduction

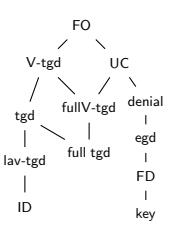
Formal argumentation serves as a widely applied framework for modeling and evaluating arguments and their reasoning, finding application in various contexts. In particular, Dung's abstract argumentation framework (Dung 1995) has been specifically designed to model conflict relationships among arguments. An abstract argumentation framework (AF) represents arguments and their conflicts through directed graphs and allows for a convenient exploration of the conflicts at an abstract level. The semantics for AFs is described in terms of sets of arguments (called extensions) that can be simultaneously accepted in a given framework.

A related yet distinct domain, with its primary focus on addressing inconsistent information, involves repairing knowledge bases and consistent query answering (CQA) (Arenas, Bertossi, and Chomicki 1999a; Chomicki 2007; Leopoldo and Bertossi 2011; Bienvenu and Rosati 2013). The goal there is to identify and *repair* inconsistencies in the data and obtain a consistent knowledge base (KB) that satisfies the imposed constraints. The current research on repairs and the theory of argumentation exhibit some overlaps. Indeed, both (CQA for inconsistent KBs and instantiated argumentation theory) address reasoning under inconsistent information (Croitoru and Vesic 2013).

While the connection between AFs and inconsistent databases has gained significant attention, no prior work has investigated the *precise expressive power* of Dung's theory of argumentation in terms of integrity constraints (ICs). In this paper, we take on this challenge and resolve the exact expressive power of Dung's AFs. This is achieved by expressing AFs in terms of inconsistent databases where ICs include functional (FDs) and inclusion dependencies (IDs). As extensions in an AF are subsets of arguments satisfying a semantics, the association with relational databases is clarified through subset-repairs (Chomicki and Marcinkowski 2005). Precisely, arguments are seen as database tuples, which together with ICs model the arguments interaction.

We *complete the mutual relationship* between inconsistent databases and argumentation frameworks in Dung's setting, thereby strengthening the connection between the two formalisms as anticipated earlier (Mahmood et al. 2024). Then, the conflict relationship between arguments closely resembles the semantics of functional dependencies, while the defense/support relation mirrors that of inclusion dependencies. We establish that *AFs can be seen as inconsistent databases*, as so far only the converse has already been established (Bienvenu and Bourgaux 2020; Mahmood et al. 2024). This strong connection offers a *tabular* representation of the *graphical* AFs and demonstrates that FDs and IDs alone suffice to encode argument interactions in an AF.

In relational databases, the prominent notions of repairs include set-based repairs (Arenas, Bertossi, and Chomicki 1999b; Chomicki and Marcinkowski 2005; ten Cate, Fontaine, and Kolaitis 2012), attribute-based repairs (Wijsen 2002), and cardinality-based repairs (Lopatenko and Bertossi 2007). Here, we focus on subset-repairs. A *subset-repair of* an *inconsistent database* is obtained by removing tuples from the original database such that integrity constraints are no longer violated. It is known (Chomicki and Marcinkowski 2005) that for large classes of constraints (FDs and denial constraints), the restriction to deletions suffices to remove integrity violations. Subset-maximality is employed in this setting to assure minimal tuple removals.



Generic ICs (FO)	$\forall \boldsymbol{x}(\varphi(\boldsymbol{x}) \wedge \beta(\boldsymbol{x}) \rightarrow \bigvee_{i=1}^{n} \exists \boldsymbol{y}_i \psi_i(\boldsymbol{x}, \boldsymbol{y}_i))$
V-tgd	FO, where β is empty
tgd	V-tgd, where $n = 1$
lav-tgd	tgd, where φ is a single atom
ID	lav-tgd, where ψ_1 is a single atom
UC	FO with no existential quantifier
denial	UC, where ψ_i are empty ($n = 0$)
egd	denial, where β is a single inequality
FD	denial, where φ uses one relation symbol

Figure 1: Hierarchy of ICs and syntactic form for most commonly studied constraints (Arming, Pichler, and Sallinger 2016). Formulas φ and ψ_i are conjunctions of database atoms and β is a formula using only (in)equality symbols.

Finally, we *introduce a new family* of (attribute-based) database repairs based on a pre-existing principle of maximal content preservation. The prior research on attribute-based repairs focuses on updating attribute values (Wijsen 2002; Flesca, Furfaro, and Parisi 2005; Bertossi et al. 2008) and does not consider the setting of subset-repairs. We define the notion of covering repairs that maximally (fully) preserve the attribute values in a database. In other words, repairs that encompass a larger set of values for their attributes are preferred among all subset-repairs. We propose this novel family of repairs as a topic of independent interest. The relevance and practical implications of these repairs are underlined by connecting them to various AF semantics.

Contributions. Proof details (marked with $\star$) are available online (Mahmood, Hecher, and Ngomo 2024). In details, we establish the following (see Table 1 for an overview).

- We present a database view for Dung's theory of argumentation and prove that an AF can be seen as an inconsistent database in the presence of functional and inclusion dependencies. This also establishes the exact expressive power of AFs in terms of integrity constraints.
- We prove that the extensions of an AF correspond precisely to the subset-repairs of the resulting database for conflict-free, admissible, naive and preferred semantics.
- We propose a new family of subset-repairs based on *maximal content preservation*. While being of independent interest, they further tighten the connection of AFs to databases for stable, semi-stable and stage semantics.

Naturally, an AF (A, R) can be seen as a database with a unary relation A (arguments) and a binary relation R (attacks). Our main contributions indicate that AFs are as low in expressive power as DBs *with only FDs and IDs*. Both types of constraints are located at the lower ends of ICs hierarchy in terms of expressive power as highlighted in Figure 1. Interestingly, while the attack relation fully characterizes an AF, FDs alone can only capture this as a conflict. Fundamentally, FDs are less expressive than the attack relation and require additional support. We provide this support by IDs, which can represent the defense relation between arguments but not the conflicts. Our reductions *broaden the applicability* of systems based on evaluating DBs with ICs, e.g., (Dixit and Kolaitis 2019; Kolaitis, Pema, and Tan 2013). Since conjunctive queries offer a powerful tool for analyzing databases, a DB perspective on AFs enables fine-grained reasoning. This enables *queries beyond* extension existence or credulous/skeptical reasoning, a topic which has been motivated earlier (Dvořák, Szeider, and Woltran 2012).

Resolving the expressivity of argumentation frameworks also has a wider impact. In the argumentation community, extensions and generalizations for AFs are actively proposed and researched, such as acceptance conditions in terms of constraints (Coste-Marquis et al. 2006, Alfano et al. 2021) or abstract dialectical frameworks (Brewka and Woltran 2010). Our results, indicating that AFs have *limited expressive power*, underline the importance of this research area. Moreover, as we will show, stable, semi-stable, and stage semantics maximize certain aspects (range) of accepted arguments. This allows to define repairs maximizing certain attribute values for databases. Such repairs introduce a *set-level preference* between repairs (based on data coverage), which has not been considered before.

Related Works. AFs have been explored extensively for reasoning with inconsistent KBs (Cayrol 1995; Vesic and van der Torre 2012; Arioua, Croitoru, and Vesic 2017; Yun, Vesic, and Croitoru 2020; Bienvenu and Bourgaux 2020) and explaining query answers (Arioua et al. 2014; Arioua, Tamani, and Croitoru 2015; Hecham et al. 2017). The common goal involves formally establishing a connection between KBs and AFs, thus proving the equivalence of extensions to the repairs for a KB. This yields an *argumentative view* of the inconsistent KBs and allows implementing the CQA-semantics via AFs. FDs and IDs are two most commonly studied ICs in databases (Chomicki and Marcinkowski 2005; Livshits, Kimelfeld, and Roy 2020). The translation from DBs to AFs are known, proving that subset-repairs for an inconsistent database with FDs and IDs align with preferred extensions in the resulting AF (Mahmood et al. 2024). Finally, König, Rapberger, and Ulbricht (2022) provided several translations between Assumption Based Argumentation (Bondarenko et al. 1997), Claim-Augmented Frameworks (Dvořák and Woltran 2020a), and Argumentation Frameworks with Collective Attacks (Nielsen and Parsons 2006). We contribute to this line of research by providing translations from AFs to DBs.

Preliminaries

In the following, we briefly recall the relevant definitions.

Abstract Argumentation. We use Dung's argumentation framework (Dung 1995) and consider non-empty finite sets of arguments A. An *(argumentation) framework (AF)* is a directed graph $\mathcal{F} = (A, R)$, where A is a set of arguments and the relation $R \subseteq A \times A$ represents direct attacks between arguments. Let $S \subseteq A$, an argument $a \in A$ is *defended by S in $\mathcal{F}$*, if for every $(b, a) \in R$ there exists $c \in S$ such that $(c, b) \in R$. For $a, b \in A$ such that $(b, a) \in R$, we denote by $\mathsf{def}_b(a) := \{c \mid (c, b) \in R\}$ the set of arguments defending a against the attack by b. The characteristic function $\mathsf{D}_{\mathcal{F}}(S) \colon 2^A \rightarrow 2^A$ of $\mathcal{F}$ defined as $\mathsf{D}_{\mathcal{F}}(S) := \{a \mid a \in A, a \text{ is defended by } S \text{ in } \mathcal{F}\}$ assigns each set the arguments it defends. The *degree* of an argument

σ	Repairs	Maximality	ICs	Table Size	FDs/IDs Size	Refs.
conf	repairs	—	FDs	$\|A\| \times (\|A\|+1)$	$\|A\|$	Thm. 5/8
naive	m-repairs	Subset	FDs	$\|A\| \times (\|A\|+1)$	$\|A\|$	Thm. 5/8
adm	repairs	—	FDs+IDs	$\|A\| \times 3(\|A\|+1)$	$\|A\|/(\|A\|+1)$	Thm. 16/17
pref	m-repairs	Subset	FDs+IDs	$\|A\| \times 3(\|A\|+1)$	$\|A\|/(\|A\|+1)$	Thm. 16/17
stab	fc-repairs	Full-covering	FDs+ID	$\|A\| \times (2\|A\|+3)$	$\|A\|/1$	Thm. 21/22
stag	mc-repairs	Max-covering	FDs+ID	$\|A\| \times (2\|A\|+3)$	$\|A\|/1$	Thm. 21/22
semi-st	mc-repairs	Max-covering	FDs+IDs	$\|A\| \times 3(\|A\|+1)$	$\|A\|/(\|A\|+1)$	Thm. 21/17

Table 1: Overview of our main contributions. The AF-semantics σ (column-I) corresponds to repairs (column-II) with the type of maximality imposed (column-III), followed by integrity constraints needed to simulate AF-semantics (column-IV), the size of the resulting database tables (column-V) and the size of sets of ICs (column-VI), and references to the proofs (column-VII). The size of the database is represented as the (number of rows)×(number of columns) for number of arguments $|A|$ in the AF.

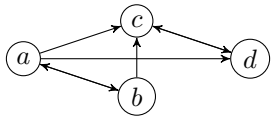


Figure 2: Argumentation framework from Example 1.

$a \in A$ is the number of arguments attacking a or attacked by a. The *degree of* $\mathcal{F}$ is the maximum degree of any $a \in A$.

In abstract argumentation one is interested in computing *extensions*, which are subsets $S \subseteq A$ of the arguments that have certain properties. The set S of arguments is called *conflict-free in* $\mathcal{F}$ if $(S \times S) \cap R = \emptyset$. Let S be conflict-free, then S is *naive in* $\mathcal{F}$ if no $S' \supset S$ is *conflict-free* in $\mathcal{F}$; *admissible in* $\mathcal{F}$ if every $a \in S$ is *defended by* S *in* $\mathcal{F}$. Further, let $S_R^+ := S \cup \{\, a \mid (b,a) \in R, b \in S \,\}$ and S be admissible. Then, S is *complete in* $\mathcal{F}$ if $\mathsf{D}_{\mathcal{F}}(S) = S$; *preferred in* $\mathcal{F}$, if no $S' \supset S$ is *admissible in* $\mathcal{F}$; *semi-stable in* $\mathcal{F}$ if no admissible set $S' \subseteq A$ in $\mathcal{F}$ with $S_R^+ \subsetneq (S')_R^+$ exists; and *stable in* $\mathcal{F}$ if every $b \in A \setminus S$ is *attacked* by some $a \in S$. Finally, a conflict-free set S is *stage in* F if there is no conflict-free $S' \subseteq A$ in F with $S_R^+ \subsetneq (S')_R^+$. For each semantic $\sigma \in \{\mathsf{naive}, \mathsf{adm}, \mathsf{comp}, \mathsf{pref}, \mathsf{semi\text{-}st}, \mathsf{stab}, \mathsf{stag}\}$, $\sigma(\mathcal{F})$ denotes the set of *all extensions* of semantics σ in $\mathcal{F}$.

Example 1. *Consider $\mathcal{F} = (A, R)$ depicted in Fig. 2. Then, $\sigma(\mathcal{F}) = \{\{b,d\},\{a\}\}$ for $\sigma \in \{\mathsf{pref}, \mathsf{stab}, \mathsf{comp}\}$. Further, $\mathsf{conf}(\mathcal{F}) = \mathsf{pref}(\mathcal{F}) \cup \{\{x\} \mid x \in A\} \cup \{\emptyset\}$, $\mathsf{naive}(\mathcal{F}) = \mathsf{pref}(\mathcal{F}) \cup \{\{c\}\}$, and $\mathsf{adm}(\mathcal{F}) = \{\{b,d\},\{b\},\{a\},\emptyset\}$.*

Databases and Repairs. For our setting, an instance of a *database (DB)* is a single table denoted as T since it suffices to prove the connection to AFs. Each entry in the table is called a *tuple* which is associated with a unique identifier (depicted in boldface $\mathbf{t} \in T$). Formally, a table corresponds to a relational schema denoted as $T(x_1, \ldots, x_n)$, where T is the relation name and $x_1, \ldots, x_n$ are distinct attributes. We denote individual attributes by small letter (e.g., x, y) and reserve capital letters (X, Y) for sequences of attributes. For an attribute x and tuple $\mathbf{s} \in T$, $\mathbf{s}[x]$ denotes the value taken by $\mathbf{s}$ for the attribute x and for a sequence $X = (x_1, \ldots, x_k)$, $\mathbf{s}[X]$ denotes the sequence $(\mathbf{s}[x_1], \ldots, \mathbf{s}[x_k])$. For an instance T, $\mathsf{dom}(T)$ denotes the *active domain* of T, i.e., the collection of all values occurring in any tuple in T. The size of a table T with m tuples and n attributes is $m \times n$.

Let $T(x_1, \ldots, x_n)$ be a schema and T be a database. In the following, we employ commonly used definitions for FDs and IDs. A *functional dependency* (FD) over T is an expression of the form $X \to Y$ for sequences X, Y of attributes in T. A database T satisfies $X \to Y$, denoted as $T \models (X \to Y)$ if for all $\mathbf{s}, \mathbf{t} \in T$: if $\mathbf{s}[X] = \mathbf{t}[X]$ then $\mathbf{s}[Y] = \mathbf{t}[Y]$. That is, every pair of tuples from T that agree on their values for attributes in X also agree on their values for Y. Moreover, an *inclusion dependency* (ID) is an expression of the form $X \subseteq Y$ for two sequences X and Y of attributes with same length. Then, T satisfies $X \subseteq Y$ ($T \models X \subseteq Y$) if for each $\mathbf{s} \in T$, there is some $\mathbf{t} \in T$ such that $\mathbf{s}[X] = \mathbf{t}[Y]$. Let $i := X \subseteq Y \in I$ be an ID and $\mathbf{s} \in T$, we say that a tuple $\mathbf{t} \in T$ *supports* $\mathbf{s}$ for the ID i if $\mathbf{s}[X] = \mathbf{t}[Y]$. We denote by $\mathsf{sup}_i(\mathbf{s}) := \{\mathbf{t} \mid \mathbf{t}[Y] = \mathbf{s}[X]\}$ the collection of tuples *supporting* $\mathbf{s}$ for i.

Let T be a database and B be a collection of FDs and IDs. T is *consistent* with respect to B, denoted as $T \models B$, if $T \models b$ for each $b \in B$. Further, T is *inconsistent* with respect to B if there is some $b \in B$ such that $T \not\models b$. A *subset-repair* of T with respect to B is a subset $P \subseteq T$ which is consistent with respect to B. Moreover, P is a *maximal repair* if there is no set $P' \subseteq T$ such that P' is also consistent with respect to B and $P \subset P'$. Note that our relaxed definition clarifies the link to AF-semantics without requiring maximality although one can substitute (our) *repairs* with *sub-repairs* to reserve *repairs* for *subset-maximal repairs*. We speak of a repair when we intend to mean a subset-repair. Let $\mathcal{D} = \langle T, D \rangle$ where T is a database and D is a set of dependencies, then $\mathsf{repairs}(\mathcal{D})$ (resp., $\mathsf{m\text{-}repairs}(\mathcal{D})$) denotes the set of all (maximal) repairs for $\mathcal{D}$. Slightly abusing the notation, we call a table T and an instance $\langle T, D \rangle$, a database.

Example 2. *Consider $\mathcal{D} = \langle T, D \rangle$ with database $T = \{\mathbf{s}_i \mid i \leq 6\}$ as depicted in Table 2, FD $f := \texttt{Tutor}, \texttt{Time} \to \texttt{Room}$, and ID $\texttt{Advisor} \subseteq \texttt{Tutor}$. $\mathcal{D}$ is inconsistent since $\{\mathbf{s}_1, \mathbf{s}_2\} \not\models f, \{\mathbf{s}_3, \mathbf{s}_4\} \not\models f$ and there is no $\mathbf{s}_i \in T$ with $\mathbf{s}_6[\texttt{advisor}] = \mathbf{s}_i[\texttt{tutor}]$. A subset containing exactly one tuple from each set $\{\mathbf{s}_1, \mathbf{s}_2\}$, $\{\mathbf{s}_3, \mathbf{s}_4\}$ and $\{\mathbf{s}_5\}$ is a maximal repair for $\mathcal{D}$. Further, $\{\mathbf{s}_1\}, \{\mathbf{s}_4\}, \{\mathbf{s}_1, \mathbf{s}_3\}$ are repairs but not maximal.*

A DB View of Abstract Argumentation

In this section, we connect inconsistent databases and abstract argumentation frameworks. This is established by

T	Tutor	Time	Room	Course	Advisor
$\mathbf{s}_1$	Alice	TS-1	A10	Logic-I	Alice
$\mathbf{s}_2$	Alice	TS-1	B20	Algorithms	Carol
$\mathbf{s}_3$	Bob	TS-2	B20	Statistics	Alice
$\mathbf{s}_4$	Bob	TS-2	C30	Calculus	Bob
$\mathbf{s}_5$	Carol	TS-3	C30	Calculus	Bob
$\mathbf{s}_6$	Carol	TS-3	B20	Algorithms	Dave

Table 2: An inconsistent database.

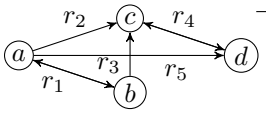
proving that an AF $\mathcal{F}$ can be seen as an instance $\mathcal{D}_\mathcal{F}$ of inconsistent database such that the acceptable sets of arguments in $\mathcal{F}$ correspond precisely to (maximal) repairs for $\mathcal{D}$. Intuitively, our construction relies on the fact that the attack relation between arguments in $\mathcal{F}$ can be simulated via a database together with FDs, and defending arguments can be seen as another database with IDs. Then, combining the two databases and dependencies, we obtain an AF — seen as an inconsistent database. For convenience, an argument "a" seen as a tuple in the database is denoted by "$\mathbf{a}$" to highlight that $\mathbf{a}$ depicts a tabular representation of a.

We first simulate the conflicts between arguments in an AF $\mathcal{F}$ via FDs, resulting in a *conflict database*. Then, we establish that the idea of *defending* arguments can be simulated by a *defense* database and IDs. Finally, we combine the two databases thereby proving that in AFs, the semantics $\sigma \in \{\mathsf{conf}, \mathsf{naive}, \mathsf{adm}, \mathsf{pref}\}$ can be simulated via (maximal) repairs for inconsistent databases with FDs and IDs. In addition, we establish the size and the time complexity associated with the construction of the aforementioned databases. For simplicity, we first assume AFs without self-attacking arguments. At the end, we highlight the changes required to allow self-attacking arguments in AFs.

Conflict DBs via Functional Dependencies

Let $\mathcal{F} = (A, R)$ be an AF with a set A of arguments and attacks R. Then, we construct a *conflict* database $\mathcal{C}_\mathcal{F} = \langle T, F\rangle$ where each argument $a \in A$ is seen as the tuple $\mathbf{a}$ (the unique identifier to the tuple representing the argument a) and a collection $F = \{f_i \mid r_i \in R\}$ of FDs. A tuple $\mathbf{a} \in T$ encodes the conflicts between $a \in A$ and other arguments in $\mathcal{F}$. Intuitively, $\mathcal{C}$ is constructed in such a way that $\{\mathbf{a}, \mathbf{b}\} \not\models f_i$ for the attack $r_i = (a, b) \in R$.

Notice that, $\{\mathbf{a}, \mathbf{b}\} = \{\mathbf{b}, \mathbf{a}\}$ is true for any $\mathbf{a}, \mathbf{b} \in T$, although the attacks (a, b) and (b, a) are not the same. As a consequence, the conflict database for an AF will be *symmetric* by design. That is, both attacks $(a, b), (b, a) \in R$ are encoded in the conflict database in the same way by requiring that $\{\mathbf{a}, \mathbf{b}\} \not\models f$ for some $f \in F$. It is worth remarking that functional dependencies can only simulate the symmetric attacks. In particular, the input AF may contain non-symmetric attacks though our translation (for the conflict database) will treat them as symmetric ones. This is unproblematic in this particular setting, since only conflict-free (naive) extensions correspond to repairs (maximal repairs) with FDs. Further, we will expound on this situation later that one can (perhaps) not simulate directed attacks by FDs alone, and requires IDs for that purpose.



T	x_1	x_2	x_3	x_4	x_5	n
$\mathbf{a}$	r_1	r_2	a	a	r_5	a
$\mathbf{b}$	r_1	b	r_3	b	b	b
$\mathbf{c}$	c	r_2	r_3	r_4	c	c
$\mathbf{d}$	d	d	d	r_4	r_5	d

Figure 3: The AF $\mathcal{F}$ (left) from Ex. 4 and a conflict database (right) modeling $\mathcal{F}$.

In the following, each attack $r \in \{(a, b), (b, a)\}$ is depicted as a set $r := \{a, b\}$.

Definition 3 (Conflict Database)**.** *Let $\mathcal{F} = (A, R)$ be an AF with arguments A and attacks R. Then, $\mathcal{C}_\mathcal{F} := \langle T, F\rangle$ defines the* conflict database *for $\mathcal{F}$, specified as follows.*

- *The attributes of T are $\{x_i, n \mid r_i \in R\}$ and we call n the* name *attribute.*
- *$F := \{x_i \to n \mid r_i \in R\}$ is the collection of FDs and each $f \in F$* corresponds *to an attack $r \in R$.*
- *$T := \{\mathbf{a} \mid a \in A\}$. Further, (1) $\mathbf{a}[n] = a$ for each $a \in A$, (2) for $r_i = \{a, b\}$: $\mathbf{a}[x_i] = r_i = \mathbf{b}[x_i]$, and (2) for $\mathbf{c} \in T$ and attributes x_j not already assigned (when all the attacks have been considered): $\mathbf{c}[x_j] = c$.*

The following example presents the conflict database for the AF $\mathcal{F} = (A, R)$ from our running example (Ex. 1).

Example 4. *For brevity, we rename each attack in $\mathcal{F}$ as depicted in Fig. 3. As discussed, the attacks $(a, b), (b, a) \in R$ are modeled as one conflict $r_1 := \{a, b\}$. The conflict database for $\mathcal{F}$ is $\mathcal{C}_\mathcal{F} = \langle T, F\rangle$, where $F = \{x_i \to n \mid i \leq 5\}$ and T as specified in Figure 3.*

Let $S \subseteq A$, then by $S_T \subseteq T$ we denote the set of tuples corresponding to arguments in S, defined as $S_T := \{\mathbf{a} \mid \mathbf{a} \in T, a \in S\}$. The following theorem establishes the relation between the extensions and repairs of the conflict database.

Theorem 5 ($\star$)**.** *Let $\mathcal{F}$ be an AF without self-attacking arguments and $\mathcal{C}_\mathcal{F}$ be its corresponding conflict database. Then, for every $S \subseteq A$, S is conflict-free (resp., naive) in $\mathcal{F}$ iff $S_T \subseteq T$ is a repair (maximal) for $\mathcal{C}_\mathcal{F}$.*

Notice that the above construction fails in general when $\mathcal{F}$ contains self-attacking arguments. That is, if $(a, a) \in R$, the only argument participating in a conflict is a but the database consisting of a singleton tuple $\{\mathbf{a}\}$ satisfies each FD trivially. We will see that this issue can be resolved by allowing an ID since a singleton tuple can also fail IDs (see Ex. 2).

Observe that the construction of conflict database from an AF includes an individual attribute x_r modeling $r \in R$. We argue that fewer attributes and FDs suffice because they can be reused, as depicted in the example below.

Example 6. *Table 3 depicts a compact representation of the conflict database from Example 4 with only three attributes $\{x_1, x_2, x_3\}$ and FDs $\{x_i \to n \mid i \leq 3\}$.*

The following lemma establishes that the size of FDs can be determined by the maximum degree of the input AF.

Lemma 7 ($\star$)**.** *Let $\mathcal{F}$ be an AF of degree γ. Then, there is a conflict database $\mathcal{C}_\mathcal{F} := \langle T, F\rangle$ for $\mathcal{F}$ such that $|F| \leq \gamma+1$.*

T	x_1	x_2	x_3	n
$\mathbf{a}$	r_1	r_5	r_2	a
$\mathbf{b}$	r_1	r_3	b	b
$\mathbf{c}$	r_4	r_3	r_2	c
$\mathbf{d}$	r_4	r_5	d	d

Table 3: A compact representation of the conflict DB (Ex. 6)

T	u_a	v_a	u_b	v_b	u_c	v_c	u_d	v_d
$\mathbf{a}$	0	0	b	b	c	c	d	d
$\mathbf{b}$	a	a	0	0	c	c	0	0
$\mathbf{c}$	a	0	b	0	0	0	d	d
$\mathbf{d}$	a	0	0	0	c	c	0	0

Table 4: The defense database for the AF $\mathcal{F}$ in Example 10.

As a consequence of Lemma 7, the following claim holds.

Theorem 8 ($\star$)**.** *Let $\mathcal{F}$ be an AF without self-attacking arguments. Then, its corresponding conflict database $\mathcal{C}_\mathcal{F}$ can be constructed in polynomial time in the size of $\mathcal{F}$. Additionally, $\mathcal{C}_\mathcal{F}$ has a table of size $|A| \times (|A| + 1)$, and uses $|A|$-many FDs at most, for an AF with $|A|$ arguments.*

Defense DBs via Inclusion Dependencies

Assume an AF $\mathcal{F} = (A, R)$ with arguments A and attacks R. We construct a *defense* database $\mathcal{D}_\mathcal{F} = \langle T, I\rangle$ for $\mathcal{F}$. Intuitively, the defense database T contains the information about incoming and outgoing attacks for each argument $a \in A$. The attributes of T are $\{u_a, v_a \mid a \in A\}$, further $T := \{\mathbf{a} \mid a \in A\}$ and the tuple $\mathbf{a} \in T$ encodes the neighborhood of the argument $a \in A$. Formally, T is defined as follows.

Definition 9 (Defense database)**.** *Let $\mathcal{F} = (A, R)$ be an AF with arguments A and attacks R. Then $\mathcal{D}_\mathcal{F} := \langle T, I\rangle$ defines the* defense *database for $\mathcal{F}$, specified in the following.*

- *The attributes of T are $\{u_a, v_a \mid a \in A\}$.*
- *$I := \{u_a \subseteq v_a \mid a \in A\}$ is the collection of IDs.*
- *$T := \{\mathbf{a} \mid a \in A\}$, where (1) for each $r = (a, b) \in R$:*

$$\mathbf{a}[u_b] = b, \quad \mathbf{a}[v_b] = b, \quad \mathbf{b}[u_a] = a, \quad \mathbf{b}[v_a] = 0.$$

(2) for each $\mathbf{t} \in T$ and variables u_d, v_d not yet assigned:

$$\mathbf{t}[u_d] = 0, \qquad \mathbf{t}[v_d] = 0.$$

Intuitively, for each $\mathbf{a} \in T$: $\mathbf{a}[v_b]$ encodes whether the argument a attacks b, by setting $\mathbf{a}[v_b] = b$ when $(a, b) \in R$. Moreover, $\mathbf{a}[u_b]$ aims at encoding whether a *interacts* with (either attacks, or attacked by) b. This is achieved by setting $\mathbf{a}[u_b] = b$ if $(b, a) \in R$ or $(a, b) \in R$. The intuition is to simulate attacks from an argument a via an ID $u_a \subseteq v_a$. The idea is that if an argument b is attacked by a, then b interacts with a and hence $\mathbf{b}[u_a] = a$. Now, there are two ways b can be defended against the attack by a: either b defends itself by attacking a (in which case $\mathbf{b}[v_b] = a$, hence $\{\mathbf{b}\} \models u_a \subseteq v_a$), or there is an argument $c \notin \{a, b\}$ attacking a (therefore $\mathbf{c}[v_a] = a$, and $\{\mathbf{b}, \mathbf{c}\} \models u_a \subseteq v_a$ as $\mathbf{c}[u_a] = \mathbf{c}[v_a] = a$).

The following example demonstrates the defense database for the AF $\mathcal{F} = (A, R)$ from our running example (Ex. 1).

Example 10. *The defense database for $\mathcal{F}$ from Example 1 is $\mathcal{D}_\mathcal{F} = \langle T, I\rangle$, where $I = \{u_s \subseteq v_s \mid s \in A\}$ and T is specified in Table 4.*

Observe that the active domain of T consists of the arguments in $\mathcal{F}$, as well as, an auxiliary element 0 to serve the purpose of missing values in the database (i.e., when a value is missing, we prefer writing 0 rather than leaving it blank). Moreover, repeating the argument names instead of 0 as for the case of the conflict database has an undesired effect. Consider Example 10, if we set $\mathbf{a}[u_a] = \mathbf{a}[v_a] = a$ instead of 0, then $\mathbf{a} \in \mathsf{sup}_i(\mathbf{d})$ and $\mathbf{a} \in \mathsf{sup}_i(\mathbf{c})$ where $i = u_a \subseteq v_a$. However, the argument a does not defend c or d in $\mathcal{F}$ against the attack by $a \in A$. Indeed, we aim at proving that the support relation between tuples in DBs for IDs is essentially the same as the defense relation between arguments. This connection is clarified in the following lemma.

Lemma 11 ($\star$)**.** *Let $a, b \in A$ be two arguments such that $(a, b) \in R$. Then, for $\mathbf{b} \in T$ (the tuple corresponding to b) and $i := u_a \subseteq v_a \in I$, we have $\mathsf{def}_a(b) = \mathsf{sup}_i(\mathbf{b})$.*

Let $S \subseteq A$ be a set of arguments and $S_T \subseteq T$ be the corresponding set of tuples. The defense database has the property that $S_T \models u_a \subseteq v_a$ iff each argument in S is defended against the argument $a \in A$. We prove that a set $S \subseteq A$ defends itself in $\mathcal{F}$ iff $S_T \subseteq T$ satisfies every ID in I.

Lemma 12 ($\star$)**.** *Let $\mathcal{F}$ be an AF and $\mathcal{D}_\mathcal{F}$ denotes its corresponding defense database. Then, for every $S \subseteq A$, S defends itself in $\mathcal{F}$ iff $S_T \subseteq T$ is a repair for $\mathcal{D}_\mathcal{F}$.*

Note that repairs for a defense database do not yield conflict-free sets. In fact, we will prove later that one can not model conflict-freeness via inclusion dependencies alone.

Example 13. *Consider the AF $\mathcal{F}$ and its defense database $\mathcal{D}_\mathcal{F}$ from Example 10. Then, $\{\mathbf{a}\}, \{\mathbf{b}\} \in \mathsf{repairs}(\mathcal{D}_\mathcal{F})$ and each set defends itself whereas $\{\mathbf{c}\}, \{\mathbf{d}\} \notin \mathsf{repairs}(\mathcal{D}_\mathcal{F})$ and do not defend themselves. Further, $\{\mathbf{a}, \mathbf{b}, \mathbf{c}\} \in \mathsf{repairs}(\mathcal{D}_\mathcal{F})$ and defends itself, although it is not conflict-free in $\mathcal{F}$.*

The following theorem establishes the size and the time bounds to construct a of defense database for a given AF.

Theorem 14 ($\star$)**.** *Let $\mathcal{F}$ be an AF without self-attacking arguments. Then, its corresponding defense database $\mathcal{D}_\mathcal{F}$ can be constructed in polynomial time in the size of $\mathcal{F}$. Additionally, $\mathcal{D}_\mathcal{F}$ has a table of size $|A| \times 2|A|$ and uses $|A|$-many IDS, for an AF with $|A|$ arguments.*

Inconsistent Databases for AFs

We combine the (conflict and defense) databases from the previous two subsections and establish that a collection of FDs and IDs suffices to encode the entire AF. Let $\mathcal{F} = (A, R)$ be an AF. We construct an *AF-database* as the instance $\mathcal{A}_\mathcal{F} = \langle T, D\rangle$, where T is the database obtained by *combining* the conflict and defense databases, and $D = F \cup I$ consists of a collection of FDs and IDs. Specifically, $\mathcal{A}_\mathcal{F}$ has the following components.

- $F := \{x_i \to n \mid r_i \in R\}$ and $I := \{u_a \subseteq v_a \mid a \in A\}$.
- $T := \{\mathbf{a} \mid a \in A\}$ is a database over attributes $\{x_i \mid r_i \in R\} \cup \{n\} \cup \{u_a, v_a \mid a \in A\}$. The tuples of T are

T	x_1	x_2	x_3	n	u_a	v_a	u_b	v_b	u_c	v_c	u_d	v_d
a	r_1	r_5	r_2	a	0	0	b	b	c	c	d	d
b	r_1	r_3	b	b	a	a	0	0	c	c	0	0
c	r_4	r_3	r_2	c	a	0	b	0	0	0	d	d
d	r_4	r_5	d	d	a	0	0	0	c	c	0	0

Table 5: The inconsistent database for the AF $\mathcal{F}$ in Ex. 15.

specified as before. That is, (1) for each $\mathbf{a} \in T$, $\mathbf{a}[n] = a$, (2) for each $r_i = (a, b) \in R$: $\mathbf{a}[x_i] = r_i$, $\mathbf{b}[x_i] = r_i$,

$$\mathbf{a}[u_b] = b, \quad \mathbf{a}[v_b] = b, \quad \mathbf{b}[u_a] = a, \quad \mathbf{b}[v_a] = 0,$$

and (3) for each $\mathbf{c} \in T$ and variables x_i, u_d, v_d not already assigned: $\mathbf{c}[x_i] = c$, $\mathbf{c}[u_d] = 0$, and $\mathbf{c}[v_d] = 0$.

Although for better presentation we used an attribute and FD for each $r \in R$, Lemma 7 is still applicable and allows us to utilize $\gamma + 1$ many FDs when $\mathcal{F}$ has degree γ. Further, the encoding of conflicts via FDs is still symmetric, i.e., an attack of the form $r_i = \{a, b\}$ is considered for assigning attribute values for x_i's. The following example demonstrates the inconsistent database from our running example (Ex. 1).

Example 15. *The AF-database for $\mathcal{F}$ is $\mathcal{A}_\mathcal{F} = \langle T, D\rangle$, where $D = F \cup I$ with $F = \{x_i \to n \mid i \leq 3\}$ and $I = \{u_s \subseteq v_s \mid s \in A\}$, and T is specified in Table 5.*

Self-attacking arguments. A self-attacking argument can not belong to any extension. We add a single ID $i_s := u_s \subseteq v_s$ over fresh attributes $\{u_s, v_s\}$ to encode self-attacking arguments. To this aim, for each $a \in A$ such that $(a, a) \in R$: we set $\mathbf{a}[u_s] = a$ and $\mathbf{a}[v_s] = 0$ for the tuple $\mathbf{a} \in T$. Since there is no tuple $\mathbf{b} \in T$ with $\mathbf{b}[v_s] = a$, the tuples corresponding to self-attacking arguments do not belong to any repair due to i_s. Henceforth, we assume that AF-databases include the attributes $\{u_s, v_s\}$ and the ID $u_s \subseteq v_s$ for self-attacking arguments. The equivalence between repairs of the conflict database and conflict-free (naive) extensions (Thm. 5) does not require the ID i_s as the conflict-free extensions in an AF $\mathcal{F}$ and those in the AF $\mathcal{F}'$ obtained from $\mathcal{F}$ by removing self-attacking arguments, coincide.

We are ready to prove that admissible and preferred extensions for $\mathcal{F}$ coincide with repairs (max. repairs) for $\mathcal{A}_\mathcal{F}$.

Theorem 16 ($\star$)**.** *Let $\mathcal{F}$ be an AF and $\mathcal{A}_\mathcal{F}$ denotes its corresponding AF-database. Then, for every $S \subseteq A$, S is admissible (resp., preferred) in $\mathcal{F}$ iff $S_T \subseteq T$ is a repair (max. repair) for $\mathcal{A}_\mathcal{F}$.*

Regarding the size of the AF-database and the time to construct it for a given AF, we have the following result.

Theorem 17 ($\star$)**.** *Let $\mathcal{F} = (A, R)$ be an AF. Then, its corresponding AF-database $\mathcal{A}_\mathcal{F}$ can be constructed in polynomial time in the size of $\mathcal{F}$. Additionally, $\mathcal{A}_\mathcal{F}$ has a table of size and $|A| \times 3(|A| + 1)$ and uses (at most) $|A|$-many FDs and $(|A| + 1)$ IDs, for an AF with $|A|$ arguments.*

For symmetric AFs (Coste-Marquis, Devred, and Marquis 2005) (i.e., $(a, b) \in R$ iff $(b, a) \in R$ for every $a, b \in A$), the stable, preferred and naive extensions coincide. As a result, one only needs the conflict database and FDs to establish the equivalence between repairs and extensions.

Corollary 18. *Let $\mathcal{F}$ be a symmetric AF and $\mathcal{C}_\mathcal{F}$ its conflict database. Then, for $S \subseteq A$ and $\sigma \in \{\mathsf{naive}, \mathsf{stab}, \mathsf{pref}\}$, $S \in \sigma(\mathcal{F})$ iff $S_T \subseteq T$ is a subset-maximal repair for $\mathcal{C}_\mathcal{F}$.*

We conclude this section by observing that one can not simulate AFs via inconsistent databases using only one type of dependencies (FDs or IDs) under the complexity theoretic assumption that the class **P** is different from **NP**. This holds because the problem to decide the existence of a non-empty repair for a database instance comprising only FDs or IDs is in **P** (Mahmood et al. 2024), whereas the problem to decide whether an AF has a non-empty admissible-extension is **NP**-complete. The stated claim holds due to Theorem 16.

Attribute-Based Repairs & Other Semantics

Besides subset maximality, we introduce maximal repairs following the principle of *content preservation* (Wijsen 2002). The motivation behind these *attribute-based repairs* is to adjust subset-maximality by retaining maximum possible values from $\mathsf{dom}(T)$ for certain attributes of T. Let $\mathcal{D} = \langle T, D\rangle$ where T is a database with values $\mathsf{dom}(T)$ and D is a set of dependencies. Let X be a sequence of attributes in T and $P \subseteq T$, then $P[X]$ denotes the database obtained from P by restricting the attributes to X and $\mathsf{dom}(P[X])$ denotes the set of values in $P[X]$. Given $\mathcal{D}$ and a sequence X of attributes, we say that $P \subseteq T$ is a *maximally covering repair* for $\mathcal{D}$ with respect to X if $P \in \mathsf{repairs}(\mathcal{D})$ and there is no $P' \in \mathsf{repairs}(\mathcal{D})$ such that $\mathsf{dom}(P'[X]) \supset \mathsf{dom}(P[X])$. Moreover, we say that P is a *fully covering repair* for $\mathcal{D}$ with respect to X if $\mathsf{dom}(P[X]) = \mathsf{dom}(T[X])$. In other words, a maximal covering repair retains maximum possible values from $\mathsf{dom}(T[X])$, whereas a fully covering repair takes all the values from $\mathsf{dom}(T[X])$. If X contains all the attributes in T then we simply speak of maximally (fully) covering repair for $\mathcal{D}$ without specifying X. For a database instance $\mathcal{D}$ and attributes X, by $\mathsf{mc\text{-}repairs}(\mathcal{D}, X)$ (resp., $\mathsf{fc\text{-}repairs}(\mathcal{D}, X)$) we denote the set of all maximally (fully) covering repairs for $\mathcal{D}$ with respect to X.

Example 19. *In the database from Example 2, $\{\mathbf{s}_1, \mathbf{s}_3, \mathbf{s}_5\}$ and $\{\mathbf{s}_2, \mathbf{s}_3, \mathbf{s}_5\}$ are both subset-maximal as well as maximally covering repairs for $\mathcal{D}$ considering all attributes. Moreover, $\{\mathbf{s}_1, \mathbf{s}_4, \mathbf{s}_5\}$ is subset-maximal but not maximally covering (since $\{\mathbf{s}_1, \mathbf{s}_3, \mathbf{s}_5\}$ covers more values). Notice that there is no fully covering repair for $\mathcal{D}$. Nevertheless, $\{\mathbf{s}_1, \mathbf{s}_3, \mathbf{s}_5\}$ is a fully covering repair for $\mathcal{D}$ with respect to attributes $\{\texttt{Tutor}, \texttt{Time}, \texttt{Room}\}$ whereas $\{\mathbf{s}_2, \mathbf{s}_3, \mathbf{s}_5\}$ is not. Further, there is no fully covering repair for $\mathcal{D}$ w.r.t $\{\texttt{Course}\}$ or $\{\texttt{Advisor}\}$.*

A maximally covering repair is also subset-maximal for any instance $\mathcal{D}$, whereas, the reverse is not true in general. Moreover, a fully covering repair is also maximally covering, though it may not always exist. Observe that, certain attribute may not be fully covered in any repair and requiring the repairs to fully cover the domain with respect to all the attributes may not always be achievable. This motivates the need to focus on specific attributes in a database and include repairs fully covering the values in those attributes.

Stable, Semi-stable, and Stage Semantics

We prove that stable, semi-stable and stage extensions in AFs can also be seen as subset-repairs of the inconsistent DBs. This is achieved by dropping subset-maximality and utilizing the covering semantics for repairs. Notice that, the three semantics require maximality for the range S^+ for a set S of arguments. Recall that the attributes $\{v_x \mid x \in A\}$ in the defense database encode arguments attacking $x \in A$. Together with the *name* attribute (n) from the conflict database, we can encode the range of a set S of arguments via an AF-database. Let $X_r := \{v_x \mid x \in A\} \cup \{n\}$ denote the *range* attributes. The following lemma proves the relation between values taken by a set S_T of tuples under attributes X_r and the range S^+ for a set $S \subseteq A$ of arguments.

Lemma 20 ($\star$). *Let $\mathcal{F}$ be an AF and $\mathcal{A}_\mathcal{F}$ denote its AF-database. Then, for every set $S \subseteq A$ of arguments and corresponding set $S_T \subseteq T$ of tuples, $S^+ = \mathsf{dom}(S_T[X_r]) \setminus \{0\}$.*

Now, we consider the AF-database for an AF $\mathcal{F}$ with range attributes X_r to determine values in $\mathsf{dom}(T)$ covered by any repair. Since stable and stage semantics only require conflict-freeness, we drop the admissibility and hence the set of IDs, except for the ID (i_s) for self-attacking arguments.

Theorem 21 ($\star$). *Let $\mathcal{F}$ be an AF and $\mathcal{A}_\mathcal{F}$ denotes its corresponding AF-database where $\mathcal{A}_\mathcal{F} = \langle T, F \cup I \rangle$. Moreover, let i_s be the ID for self-attacking arguments, $\mathcal{C}_\mathcal{F} = \langle T, F \cup \{i_s\} \rangle$, and let $X_r = \{v_x \mid x \in A\} \cup \{n\}$. Then, for every $S \subseteq A$ and corresponding $S_T \subseteq T$,*

- $S \in \mathsf{stab}(\mathcal{F})$ *iff* $S_T \in \mathsf{fc\text{-}repairs}(\mathcal{C}_\mathcal{F}, X_r)$,
- $S \in \mathsf{stag}(\mathcal{F})$ *iff* $S_T \in \mathsf{mc\text{-}repairs}(\mathcal{C}_\mathcal{F}, X_r)$
- $S \in \mathsf{semi\text{-}st}(\mathcal{F})$ *iff* $S_T \in \mathsf{mc\text{-}repairs}(\mathcal{A}_\mathcal{F}, X_r)$

For stable and stage semantics, the size of the database can be reduced since there are no IDs involved (except i_s).

Corollary 22 ($\star$). *Let $\mathcal{F} = (A, R)$ be an AF. Then, its corresponding AF-database $\mathcal{C}_\mathcal{F}$ for stable and stage semantics has a table of size $|A| \times (2|A| + 3)$, and uses $|A|$-many FDs with a single ID.*

Discussion, Conclusion, and Future Work

We presented a database view of Dung's theory of abstract argumentation. While our results tighten the connection between two domains by simulating Dung's argumentation frameworks by inconsistent databases, they also provide the exact expressivity of AFs via integrity constraints. Roughly speaking: Dung's argumentation frameworks can be equivalently seen as inconsistent databases where ICs include functional and inclusion dependencies. This strong connection allows us to transfer further ideas from one domain to another. As already pointed out, a repairing semantics based on the idea of maximizing *range* from AFs yields maximal (full) content preservation across certain attributes. We propose this new family of repairs as a topic of interest.

Complete and Grounded Semantics. We note that the complete and grounded semantics may not have a natural counterpart in the subset-repairs setting for databases when FDs and IDs are employed. Although, we can introduce a notion of closed repairs utilizing the connection between defending arguments and supporting tuples (Lemma 11) to simulate the closure properties required by these semantics, this may not correspond to a natural definition for repairs in the databases. Then the question arises: are there classes of ICs that can express these two semantics for AFs?

Research Directions. Our work is in line with the existing literature connecting the two domains; seeing tuples in the database as arguments (Bienvenu and Bourgaux 2020; Mahmood et al. 2024) or constructing arguments from KBs (Vesic and van der Torre 2012; Croitoru and Vesic 2013). By inspecting the expressivity of AFs to integrity constraints, we believe to have opened a new research dimension within argumentation. This study can be extended by exploring the expressivity of extensions and generalizations of AFs, e.g., dialectical frameworks (ADFs) of Brewka and Woltran (2010), set-based AFs (Nielsen and Parsons 2006) and claim-centric AFs (Dvořák and Woltran 2020b) to name a few. Moreover, in claim-centric AFs, the so-called *claim-level* semantics requires maximizing accepted sets of claims rather than arguments — which can be covered by maximally covering repairs for suitable attributes. Further, the connection between probabilistic (Li, Oren, and Norman 2011) and preference-based (Kaci, van der Torre, and Villata 2018) AFs and their database-counterpart (Lian, Chen, and Song 2010; Staworko, Chomicki, and Marcinkowski 2012) also seems promising.

The well-known reasoning problems in AFs include *credulous* (resp., *skeptical*) reasoning, asking whether an argument belongs to some (every) extension. However, the natural reasoning problem for databases includes consistent query answering (CQA) under *brave* or *cautious* semantics. A (Boolean) query is entailed bravely (resp., cautiously) from an inconsistent database if it holds in some (every) repair. It is interesting to explore DB-variants of credulous and skeptic reasoning. That is, given an argument a in an AF $\mathcal{F}$ and semantics σ, construct a query q^a_σ such that a is credulously (skeptically) accepted in $\mathcal{F}$ under semantics σ iff the query q^a_σ is entailed bravely (cautiously) from the database $\mathcal{D}_\mathcal{F}$ for $\mathcal{F}$ under appropriate repairing semantics ρ_σ. Yet another interesting direction for future work is to utilize recently developed (so-called) decomposition-guided (Fichte et al. 2021, 2023; Hecher et al. 2024) reductions for reasoning in DBs. These reductions would allow to establish tight runtime bounds for CQA via the known bounds for reasoning in AFs. Exploring these directions is left for future work.

Finally, a complexity analysis for repairs involving maximally or fully covering semantics is on our agenda. Some complexity lower bounds for the decision problems (involving FDs and IDs) already follow from the results in this paper. We next aim to target the complexity of repair checking and CQA involving not only FDs and IDs, but also more general types of constraints such as *equality* and *tuple* generating dependencies. It is noteworthy that these definitions for repairs introduce a preference on sets of tuples, distinguishing them from research on prioritized databases (Staworko, Chomicki, and Marcinkowski 2012; Kimelfeld, Livshits, and Peterfreund 2017; Bienvenu and Bourgaux 2020), which typically involves tuple preferences.

Acknowledgments

The work has received funding from the Austrian Science Fund (FWF), grants J 4656 and P 32830, the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation), grants TRR 318/1 2021 – 438445824, the European Union's Horizon Europe research and innovation programme within project ENEXA (101070305), the Society for Research Funding in Lower Austria (GFF, Gesellschaft für Forschungsförderung NÖ), grant ExzF-0004, as well as the Vienna Science and Technology Fund (WWTF), grant ICT19-065, and the Ministry of Culture and Science of North Rhine-Westphalia (MKW NRW) within project SAIL, grant NW21-059D.

References

Alfano, G.; Greco, S.; Parisi, F.; and Trubitsyna, I. 2021. Argumentation frameworks with strong and weak constraints: Semantics and complexity. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, 6175–6184.

Arenas, M.; Bertossi, L.; and Chomicki, J. 1999a. Consistent query answers in inconsistent databases. In *Proceedings of the eighteenth ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems*, 68–79.

Arenas, M.; Bertossi, L.; and Chomicki, J. 1999b. Consistent query answers in inconsistent databases. In *PODS*, volume 99, 68–79. Citeseer.

Arioua, A.; Croitoru, M.; and Vesic, S. 2017. Logic-based argumentation with existential rules. *International Journal of Approximate Reasoning*, 90: 76–106.

Arioua, A.; Tamani, N.; and Croitoru, M. 2015. Query answering explanation in inconsistent datalog knowledge bases. In *International Conference on Data Management in Cloud, Grid and P2P Systems*, 203–219. Springer.

Arioua, A.; Tamani, N.; Croitoru, M.; and Buche, P. 2014. Query failure explanation in inconsistent knowledge bases using argumentation. *Computational Models of Argument: Proceedings of COMMA 2014*, 266: 101.

Arming, S.; Pichler, R.; and Sallinger, E. 2016. Complexity of repair checking and consistent query answering. *LIPIcs*, 48(2016).

Bertossi, L.; Bravo, L.; Franconi, E.; and Lopatenko, A. 2008. The complexity and approximation of fixing numerical attributes in databases under integrity constraints. *Information Systems*, 33(4-5): 407–434.

Bienvenu, M.; and Bourgaux, C. 2020. Querying and Repairing Inconsistent Prioritized Knowledge Bases: Complexity Analysis and Links with Abstract Argumentation. In *Proceedings of the International Conference on Principles of Knowledge Representation and Reasoning*, volume 17, 141–151.

Bienvenu, M.; and Rosati, R. 2013. Tractable approximations of consistent query answering for robust ontology-based data access. In *Twenty-Third International Joint Conference on Artificial Intelligence*.

Bondarenko, A.; Dung, P. M.; Kowalski, R. A.; and Toni, F. 1997. An abstract, argumentation-theoretic approach to default reasoning. *Artificial intelligence*, 93(1-2): 63–101.

Brewka, G.; and Woltran, S. 2010. Abstract Dialectical Frameworks. In Lin, F.; Sattler, U.; and Truszczynski, M., eds., *Proceedings of the 12th International Conference on Principles of Knowledge Representation and Reasoning (KR'10)*, 102–111.

Cayrol, C. 1995. On the relation between argumentation and non-monotonic coherence-based entailment. In *IJCAI*, volume 95, 1443–1448.

Chomicki, J. 2007. Consistent query answering: Five easy pieces. In *International Conference on Database Theory*, 1–17. Springer.

Chomicki, J.; and Marcinkowski, J. 2005. Minimal-change integrity maintenance using tuple deletions. *Information and Computation*, 197(1): 90–121.

Coste-Marquis, S.; Devred, C.; and Marquis, P. 2005. Symmetric Argumentation Frameworks. In Godo, L., ed., *Symbolic and Quantitative Approaches to Reasoning with Uncertainty, 8th European Conference, ECSQARU 2005, Barcelona, Spain, July 6-8, 2005, Proceedings*, volume 3571 of *Lecture Notes in Computer Science*, 317–328. Springer.

Coste-Marquis, S.; Devred, C.; and Marquis, P. 2006. Constrained Argumentation Frameworks. In *KR'06*, 112–122.

Croitoru, M.; and Vesic, S. 2013. What can argumentation do for inconsistent ontology query answering? In *International Conference on Scalable Uncertainty Management*, 15–29. Springer.

Dixit, A. A.; and Kolaitis, P. G. 2019. A SAT-based system for consistent query answering. In *Theory and Applications of Satisfiability Testing–SAT 2019: 22nd International Conference, SAT 2019, Lisbon, Portugal, July 9–12, 2019, Proceedings 22*, 117–135. Springer.

Dung, P. M. 1995. On the acceptability of arguments and its fundamental role in nonmonotonic reasoning, logic programming and n-person games. *Artificial intelligence*, 77(2): 321–357.

Dvořák, W.; Szeider, S.; and Woltran, S. 2012. Abstract argumentation via monadic second order logic. In *International Conference on Scalable Uncertainty Management*, 85–98. Springer.

Dvořák, W.; and Woltran, S. 2020a. Complexity of abstract argumentation under a claim-centric view. *Artificial Intelligence*, 285: 103290.

Dvořák, W.; and Woltran, S. 2020b. Complexity of abstract argumentation under a claim-centric view. *Artificial Intelligence*, 285: 103290.

Fichte, J. K.; Hecher, M.; Mahmood, Y.; and Meier, A. 2021. Decomposition-Guided Reductions for Argumentation and Treewidth. In *IJCAI*, volume 21, 1880–1886.

Fichte, J. K.; Hecher, M.; Mahmood, Y.; and Meier, A. 2023. Quantitative Reasoning and Structural Complexity for Claim-Centric Argumentation. In *IJCAI*, 3212–3220.

Flesca, S.; Furfaro, F.; and Parisi, F. 2005. Consistent Query Answers on Numerical Databases Under Aggregate Constraints. In Bierman, G.; and Koch, C., eds., *Database Programming Languages*, 279–294. Berlin, Heidelberg: Springer Berlin Heidelberg.

Hecham, A.; Arioua, A.; Stapleton, G.; and Croitoru, M. 2017. An empirical evaluation of argumentation in explaining inconsistency tolerant query answering. In *DL: Description Logics*, 1879.

Hecher, M.; Mahmood, Y.; Meier, A.; and Schmidt, J. 2024. Quantitative claim-centric reasoning in logic-based argumentation. In *Proceedings of the 33rd International Joint Conference on Artificial Intelligence (IJCAI'24). ijcai. org. To appear*.

Kaci, S.; van der Torre, L.; and Villata, S. 2018. Preference in Abstract Argumentation. In *Computational Models of Argument*, 405–412. IOS Press.

Kimelfeld, B.; Livshits, E.; and Peterfreund, L. 2017. Detecting ambiguity in prioritized database repairing. In *20th International Conference on Database Theory (ICDT 2017)*. Schloss-Dagstuhl-Leibniz Zentrum für Informatik.

Kolaitis, P. G.; Pema, E.; and Tan, W.-C. 2013. Efficient querying of inconsistent databases with binary integer programming. *Proceedings of the VLDB Endowment*, 6(6): 397–408.

König, M.; Rapberger, A.; and Ulbricht, M. 2022. Just a matter of perspective. In *Computational Models of Argument*, 212–223. IOS Press.

Leopoldo, B.; and Bertossi, S. 2011. Database repairing and consistent query answering. *Morgan & Claypool Publishers*.

Li, H.; Oren, N.; and Norman, T. J. 2011. Probabilistic argumentation frameworks. In *International workshop on theorie and applications of formal argumentation*, 1–16. Springer.

Lian, X.; Chen, L.; and Song, S. 2010. Consistent query answers in inconsistent probabilistic databases. In *Proceedings of the 2010 ACM SIGMOD International Conference on Management of data*, 303–314.

Livshits, E.; Kimelfeld, B.; and Roy, S. 2020. Computing optimal repairs for functional dependencies. *ACM Transactions on Database Systems (TODS)*, 45(1): 1–46.

Lopatenko, A.; and Bertossi, L. E. 2007. Complexity of Consistent Query Answering in Databases Under Cardinality-Based and Incremental Repair Semantics. In *Proceedings ICDT-2007*, volume 4353 of *LNCS*, 179–193. Springer.

Mahmood, Y.; Hecher, M.; and Ngomo, A.-C. N. 2024. Dung's Argumentation Framework: Unveiling the Expressive Power with Inconsistent Databases. arXiv:2412.11617.

Mahmood, Y.; Virtema, J.; Barlag, T.; and Ngomo, A.-C. N. 2024. Computing Repairs Under Functional and Inclusion Dependencies via Argumentation. In *International Symposium on Foundations of Information and Knowledge Systems*, 23–42. Springer.

Nielsen, S. H.; and Parsons, S. 2006. A Generalization of Dung's Abstract Framework for Argumentation: Arguing with Sets of Attacking Arguments. In Maudet, N.; Parsons, S.; and Rahwan, I., eds., *Argumentation in Multi-Agent Systems, Third International Workshop, ArgMAS 2006, Hakodate, Japan, May 8, 2006, Revised Selected and Invited Papers*, volume 4766 of *Lecture Notes in Computer Science*, 54–73. Springer.

Staworko, S.; Chomicki, J.; and Marcinkowski, J. 2012. Prioritized repairing and consistent query answering in relational databases. *Annals of Mathematics and Artificial Intelligence*, 64(2-3): 209–246.

ten Cate, B.; Fontaine, G.; and Kolaitis, P. G. 2012. On the Data Complexity of Consistent Query Answering. In *Proceedings of the 15th International Conference on Database Theory*, ICDT '12, 22–33.

Vesic, S.; and van der Torre, L. 2012. Beyond maxi-consistent argumentation operators. In *European Workshop on Logics in Artificial Intelligence*, 424–436. Springer.

Wijsen, J. 2002. Condensed Representation of Database Repairs for Consistent Query Answering. In *Proceedings of the 9th International Conference on Database Theory*, ICDT '03, 378–393. Springer-Verlag.

Yun, B.; Vesic, S.; and Croitoru, M. 2020. Sets of Attacking Arguments for Inconsistent Datalog Knowledge Bases. In Prakken, H.; Bistarelli, S.; Santini, F.; and Taticchi, C., eds., *Computational Models of Argument - Proceedings of COMMA 2020, Perugia, Italy, September 4-11, 2020*, volume 326 of *Frontiers in Artificial Intelligence and Applications*, 419–430. IOS Press.