Journal of Artificial Intelligence Research 54 (2015) 193–231 Submitted 6/15; published 10/15

Expressiveness of Two-Valued Semantics for Abstract Dialectical Frameworks

Hannes Strass STRASS@INFORMATIK.UNI-LEIPZIG.DE
Computer Science Institute, Leipzig University
Augustusplatz 10, 04109 Leipzig, Germany

Abstract

We analyse the expressiveness of Brewka and Woltran's abstract dialectical frameworks for two-valued semantics. By expressiveness we mean the ability to encode a desired set of two-valued interpretations over a given propositional vocabulary A using only atoms from A. We also compare ADFs' expressiveness with that of (the two-valued semantics of) abstract argumentation frameworks, normal logic programs and propositional logic. While the computational complexity of the two-valued model existence problem for all these languages is (almost) the same, we show that the languages form a neat hierarchy with respect to their expressiveness. We then demonstrate that this hierarchy collapses once we allow to introduce a linear number of new vocabulary elements. We finally also analyse and compare the representational succinctness of ADFs (for two-valued model semantics), that is, their capability to represent two-valued interpretation sets in a space-efficient manner.

1. Introduction

More often than not, different knowledge representation languages have conceptually similar and partially overlapping intended application areas. What are we to do if faced with an application and a choice of several possible knowledge representation languages which could be used for the application? One of the first axes along which to compare different formalisms that comes to mind is computational complexity: if a language is computationally too expensive when considering the problem sizes typically encountered in practice, then this is a clear criterion for exclusion. But what if the available language candidates have the same computational complexity? If their expressiveness in the computational-complexity sense of "What kinds of *problems* can the formalism solve?" is the same, we need a more fine-grained notion of expressiveness. In this paper, we use such a notion and study the expressiveness of abstract dialectical frameworks (ADFs) (Brewka & Woltran, 2010; Brewka, Ellmauthaler, Strass, Wallner, & Woltran, 2013), a recent generalisation of abstract argumentation frameworks (AFs) (Dung, 1995).

Argumentation frameworks are the de-facto standard formalism in abstract argumentation, a field that studies how (abstract) arguments relate to each other in terms of directed conflicts ("attacks"), and how these conflicts can be resolved without "looking into" the arguments. While AFs are popular and well-studied, it has been noted many times in the literature that their expressive capabilities are somewhat limited. This has only recently been made technically precise by Dunne, Dvořák, Linsbichler, and Woltran (2014, 2015), who basically showed that introducing new, purely technical arguments is sometimes inevitable when using AFs for representation purposes. However, due to their very nature, the dialectical meaning of such technical arguments might be – ironically – debatable.

©2015 AI Access Foundation. All rights reserved.

Not surprisingly, quite a number of generalisations of AFs have been proposed (for an overview we refer to Brewka, Polberg, & Woltran, 2014). As one of the most general AF alternatives, the aforementioned abstract dialectical frameworks (ADFs) have emerged. In that formalism, not only arguments (called statements there) are abstract, but also links between arguments. While in AFs the links are necessarily attacks, in ADFs each statement has an associated acceptance condition – a Boolean function over its parent statements – that specifies exactly when the statement can be accepted. In this way, acceptance conditions ultimately express the meaning of links in an ADF. Even the restricted subclass of *bipolar* ADFs – where intuitively all links are supporting or attacking – is a proper generalisation of AFs, and a quite expressive one as we shall see in this paper.

ADFs could be called the lovechild of AFs and logic programs, since they combine intuitions and semantics from Dung-style abstract argumentation as well as logic programming (Brewka et al., 2013; Strass, 2013; Alviano & Faber, 2015). While on the abstract level, ADFs are intended to function as "argumentation middleware" – a sufficiently expressive target formalism for translations from more concrete (application) formalisms. As part of the ADF success story, we just mention a reconstruction of the Carneades model of argument (Brewka & Gordon, 2010), an instantiation of simple defeasible theories into ADFs (Strass, 2015a), and recent applications of ADFs for legal reasoning and reasoning with cases by Al-Abdulkarim, Atkinson, and Bench-Capon (2014, 2015).

In this paper, we approach abstract dialectical frameworks as knowledge representation formalisms, since they are used to represent knowledge about arguments and relationships between these arguments. We employ this view to analyse the representational capabilities of ADFs. Due to their roots in AFs and logic programs, we also compare the representational capabilities of those formalisms in the same setting. In this initial study we restrict ourselves to looking at two-valued semantics, more specifically the ADF (stable) model semantics, which corresponds to AF stable extension semantics, and the supported and stable model semantics for logic programs. We add propositional logic to have a well-known reference point. Analysing these precise formalisms additionally makes sense to us because the computational complexity of their respective model existence problems is the same (with one exception):

- for AFs, deciding stable extension existence is NP-complete (Dimopoulos, Nebel, & Toni, 2002);
- for normal logic programs, deciding the existence of supported/stable models is NP-complete (Bidoit & Froidevaux, 1991; Marek & Truszczyński, 1991);
- for ADFs, deciding the existence of (supported) models is NP-complete (Brewka et al., 2013), deciding the existence of stable models is Σ_2^P-complete for general ADFs (Brewka et al., 2013) and NP-complete for the subclass of bipolar ADFs (Strass & Wallner, 2015);
- the propositional satisfiability problem is NP-complete.

In view of these almost identical complexities, we use an alternative measure of the expressiveness of a knowledge representation formalism $\mathcal{F}$: "Given a set of two-valued interpretations, is there a knowledge base in $\mathcal{F}$ that has this exact model set?" This notion lends itself straightforwardly to compare different formalisms (Gogic, Kautz, Papadimitriou, & Selman, 1995):

> Formalism $\mathcal{F}_2$ is at least as expressive as formalism $\mathcal{F}_1$ if and only if every knowledge base in $\mathcal{F}_1$ has an equivalent knowledge base in $\mathcal{F}_2$.

So here expressiveness is understood in terms of *realisability*, "What kinds of model sets can the formalism express?" (In model theory, this is known as *definability*.)

It is easy to see that propositional logic can express any set of two-valued interpretations, it is *universally expressive*. The same is easy (but less easy) to see for normal logic programs under supported model semantics. For normal logic programs under *stable* model semantics, it is clear that not all model sets can be expressed, since two different stable models are always incomparable with respect to the subset relation.[1] In this paper, we study such expressiveness properties for all the mentioned formalisms under different semantics. It turns out that the languages form a more or less strict expressiveness hierarchy, with AFs at the bottom, ADFs and LPs under stable semantics higher up and ADFs and LPs under supported model semantics at the top together with propositional logic.

To show that a language $\mathcal{F}_2$ is at least as expressive as a language $\mathcal{F}_1$ we will mainly use two different techniques. In the best case, we can use a syntactic compact and faithful translation from knowledge bases of $\mathcal{F}_1$ to those of $\mathcal{F}_2$. *Compact* means that the translation does not change the vocabulary, that is, does not introduce new atoms. *Faithful* means that the translation exactly preserves the models of the knowledge base for respective semantics of the two languages. In the second best case, we assume the knowledge base of $\mathcal{F}_1$ to be given in the form of a set X of desired models and construct a semantic *realisation* of X in $\mathcal{F}_2$, that is, a knowledge base in $\mathcal{F}_2$ with model set precisely X. To show that language $\mathcal{F}_2$ is *strictly more expressive* than $\mathcal{F}_1$, we additionally have to present a knowledge base kb from $\mathcal{F}_2$ of which we prove that $\mathcal{F}_1$ cannot express the model set of kb.

Analysing the expressiveness of argumentation formalisms is a quite recent strand of work. Its ascent can be attributed to Dunne et al. (2014, 2015), who studied realisability for argumentation frameworks (allowing to introduce new arguments as long as they are never accepted). Likewise, Dyrkolbotn (2014) analysed AF realisability under projection (allowing to introduce new arguments) for three-valued semantics. Baumann, Dvořák, Linsbichler, Strass, and Woltran (2014) studied the expressiveness of the subclass of "compact" AFs, where each argument is accepted at least once. Finally, and most recently, Pührer (2015) analysed the realisability of three-valued semantics for ADFs. Previous more preliminary works include that of Brewka, Dunne, and Woltran (2011), who translated ADFs into AFs for the ADF model and AF stable extension semantics, however this translation introduces additional arguments and is therefore not compact; and ours (Strass, 2013), where we studied the syntactic intertranslatability of ADFs and LPs, but did not look at expressiveness or realisability.

The gain that is achieved by our analysis in this paper is not only that of increased clarity about fundamental properties of these knowledge representation languages – *What can these formalisms express, actually?* – but has several further applications. As Dunne et al. (2015) remarked, a major application is in constructing knowledge bases with the aim

1. However, the stable model semantics becomes universally expressive once we allow nested expressions of the form "*not not p*" in rule bodies (Lifschitz, Tang, & Turner, 1999; Lifschitz & Razborov, 2006).

of encoding a certain model set. As a necessary prerequisite to this, it must be known that the intended model set is realisable in the first place. For example, in a recent approach to revising argumentation frameworks (Coste-Marquis, Konieczny, Mailly, & Marquis, 2014), the authors avoid this problem by assuming to produce a *collection* of AFs whose model sets in union produce the desired model set. While the work of Dunne et al. (2015) showed that this is indeed necessary in the case of AFs and stable extension semantics, our work shows that for ADFs under the model semantics, a single knowledge base (ADF) is always enough to realise any given model set. What is more, if we assume that the intended model set is given in the form of a propositional formula, then the size of the realising ADF is at most linear in the size of the formula. This is only one example – we will on several occasions also consider the sizes of realisations, as is not uncommon in logic-based AI (Darwiche & Marquis, 2002; Lifschitz & Razborov, 2006; French, van der Hoek, Iliev, & Kooi, 2013; Shen & Zhao, 2014). Indeed, representation size is a fundamental practical aspect of knowledge representation languages: universal expressiveness is of little use if the model sets to express require exponential-size knowledge bases even in the best case!

Of course, the fact that the languages we study have the same computational complexity means that there in principle exist polynomial intertranslations for the respective decision problems. But such intertranslations may involve the introduction of a polynomial number of new atoms. In theory, an increase from n atoms to n^k atoms for some $k > 1$ is of no consequence. In practice, it has a profound impact: the number n of atoms directly influences the search space that any implementation potentially has to cover. There, a step from 2^n to

$$2^{n^k} = 2^{n^{k-1}n} = \left(2^{n^{k-1}}\right)^n$$

amounts to an *exponential* increase in search space size. Being able to realise a model set compactly, without new atoms, therefore attests that a formalism $\mathcal{F}$ has a certain basic kind of efficiency property, in the sense that the $\mathcal{F}$-realisation of a model set does not unnecessarily enlarge the search space of algorithms operating on it.

It might seem that it is a restricting assumption to view formalisms as sets $\mathcal{F}$ of knowledge bases kb where $\mathcal{F}$ is associated with a two-valued semantics. However, this language representation model is universal in the sense that it is just another way of expressing languages as sets of words over $\{0,1\}$. Using an n-element vocabulary $A_n = \{a_1, \ldots, a_n\}$, a binary word $w = x_1 x_2 \cdots x_n$ of length n is encoded as the set $M_w = \{a_i \in A_n \mid x_i = 1\} \subseteq A_n$. For example, using the vocabulary $A_3 = \{a_1, a_2, a_3\}$, the binary word 101 of length 3 corresponds to the set $M_{101} = \{a_1, a_3\}$. Consequently, a set L_n of words of length n can be represented by a set $X_{L_n} \subseteq 2^{A_n}$ of subsets of A_n: $X_{L_n} = \{M_w \mid w \in L_n\}$. With the above example vocabulary, the word set $L_3 = \{101, 110, 011\}$ is represented by the model set $X_{L_3} = \{\{a_1, a_3\}, \{a_1, a_2\}, \{a_2, a_3\}\}$. Conversely, each sequence $(X_n)_{n \geq 0}$ of sets with $X_n \subseteq 2^{A_n}$ uniquely determines a language $L = \bigcup_{n \geq 0} L_n$ over $\{0,1\}$: for each $n \in \mathbb{N}$, we have $L_n = \{w_M \mid M \in X_n\}$ with $w_M = x_1 x_2 \cdots x_n$ where for each $i \in \{1, \ldots, n\}$, $x_i = 1$ if $a_i \in M$ and $x_i = 0$ if $a_i \notin M$. In this paper we use "language" to refer to object-level languages while "formalism" refers to meta-level languages, such as propositional logic, argumentation frameworks, abstract dialectical frameworks, and logic programs.

Formally, the syntax of ADFs is defined via Boolean functions. However, we are interested in representations of ADFs. So we have to fix a representation of ADFs via fixing a representation of Boolean functions. We choose to use (unrestricted) propositional formulas, as is customary in most of the literature (Brewka & Woltran, 2010; Brewka et al., 2013; Polberg et al., 2013; Polberg, 2014; Gaggl & Strass, 2014; Linsbichler, 2014; Strass & Wallner, 2015; Pührer, 2015; Gaggl, Rudolph, & Strass, 2015). Exceptions to this custom are the works of Brewka et al. (2011), who use Boolean circuits, and one of ours (Strass, 2013) where we used characteristic models (that is, used a representation that is equivalent to representing the formulas in disjunctive normal form). For the subclass of bipolar ADFs, yet no uniform representation exists, which is another question we address in this paper.

By propositional formulas over a vocabulary A we mean formulas over the Boolean basis $\{\wedge, \vee, \neg\}$, that is, trees whose leaves (sinks) are atoms from A or the logical constants true $\top$ or false $\bot$, and internal nodes are either unary ($\neg$) or binary ($\wedge$,$\vee$). We also make occasional use of Boolean circuits, where "trees" above is replaced by "directed acyclic graphs"; in particular, we allow unbounded fan-in, that is, reusing sub-circuits. As usual, the depth of a formula (circuit) is the length of the longest path from the root to a leaf (sink). Figure 1 below shows formula and circuit examples of depth 3.

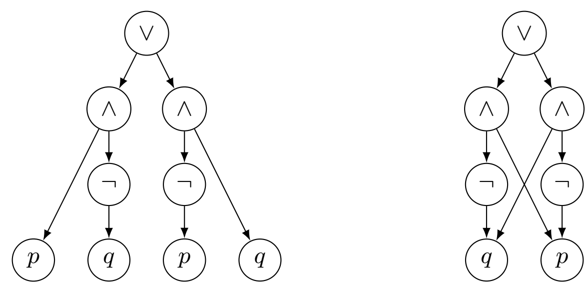


Figure 1: Representing $(p \wedge \neg q) \vee (q \wedge \neg p)$ as a formula tree (left) and a circuit (right).

Analysing the expressive power and representation size of Boolean circuits is an established sub-field of computational complexity (Arora & Barak, 2009). This has led to a number of language classes whose members can be recognised by Boolean circuits satisfying certain restrictions. We will need the class AC^0, which contains all languages $L = \bigcup_{n \geq 0} L_n$ for which there exist $d, k \in \mathbb{N}$ such that for each $n \in \mathbb{N}$, there exists a Boolean circuit C_n of depth at most d and size at most n^k where the models of C_n exactly express L_n.[2] In other words, every language $L \in \mathsf{AC}^0$ can be recognised by a family of polynomial-size Boolean circuits of a fixed maximal depth that is independent of word length.

The paper proceeds as follows. We first define the notion of expressiveness (and succinctness) formally and then introduce the formalisms we will study. After reviewing several intertranslatability results for these languages, we step-wise obtain the results that lead to the expressiveness hierarchy, while at times also looking at representational efficiency. We finally show that allowing to linearly expand the vocabulary leads to a collapse of the hierarchy. The paper concludes with a discussion of possible future work.

2. To be more precise, for each $n \in \mathbb{N}$, the models of C_n are exactly X_{L_n}, which in turn expresses L_n.

2. Background

We presume a finite set A of atoms (statements, arguments), the *vocabulary*. A knowledge representation formalism interpreted over A is then some set $\mathcal{F}$; a (two-valued) semantics for $\mathcal{F}$ is a mapping $\sigma : \mathcal{F} \to 2^{2^A}$ that assigns sets of two-valued models to knowledge bases $\mathsf{kb} \in \mathcal{F}$. (So A is implicit in σ.) Strictly speaking, a two-valued interpretation is a mapping from the set of atoms into the two truth values true and false, but for technical ease we represent two-valued interpretations by the sets containing the atoms that are true. Below, we write $\sigma(\mathcal{F}) = \{\sigma(\mathsf{kb}) \mid \mathsf{kb} \in \mathcal{F}\}$; intuitively, $\sigma(\mathcal{F})$ is the set of interpretation sets that formalism $\mathcal{F}$ can express, with any knowledge base whatsoever. For example, for $\mathcal{F} = \text{PL}$ propositional logic and $\sigma = \mathit{mod}$ the usual model semantics, we have $\sigma(\text{PL}) = 2^{2^A}$ since obviously any set of models is realisable in propositional logic.[3] This leads us to compare different pairs of languages and semantics with respect to the semantics' range of models. Our concept of "formalism" concentrates on semantics and decidedly remains abstract. We first define the expressiveness relation among formalisms.

Definition 1. Let A be a finite vocabulary, $\mathcal{F}_1, \mathcal{F}_2$ be formalisms that are interpreted over A and $\sigma_1 : \mathcal{F}_1 \to 2^{2^A}$ and $\sigma_2 : \mathcal{F}_2 \to 2^{2^A}$ be two-valued semantics. We define

$$\mathcal{F}_1^{\sigma_1} \leq_e \mathcal{F}_2^{\sigma_2} \quad \text{iff} \quad \sigma_1(\mathcal{F}_1) \subseteq \sigma_2(\mathcal{F}_2)$$

Intuitively, formalism $\mathcal{F}_2$ under semantics σ_2 is at least as expressive as formalism $\mathcal{F}_1$ under semantics σ_1, because all model sets that $\mathcal{F}_1$ can express under σ_1 are also contained in those that $\mathcal{F}_2$ can produce under σ_2. (If the semantics are clear from the context we will omit them; this holds in particular for argumentation frameworks and propositional logic, where we only look at a single semantics.) As usual,

- $\mathcal{F}_1 <_e \mathcal{F}_2$ iff $\mathcal{F}_1 \leq_e \mathcal{F}_2$ and $\mathcal{F}_2 \not\leq_e \mathcal{F}_1$;
- $\mathcal{F}_1 \cong_e \mathcal{F}_2$ iff $\mathcal{F}_1 \leq_e \mathcal{F}_2$ and $\mathcal{F}_2 \leq_e \mathcal{F}_1$.

The relation $\leq_e$ is reflexive and transitive by definition, but not necessarily antisymmetric. That is, there might different formalisms $\mathcal{F}_1 \neq \mathcal{F}_2$ that are equally expressive: $\mathcal{F}_1 \cong_e \mathcal{F}_2$.

We next introduce the succinctness relation as defined by Gogic et al. (1995).

Definition 2. Let A be a finite vocabulary; let $\mathcal{F}_1$ and $\mathcal{F}_2$ be formalisms that are interpreted over A, have size measures $\|\cdot\|_1$ and $\|\cdot\|_2$, and two-valued semantics σ_1 and σ_2, respectively. Define $\mathcal{F}_1^{\sigma_1} \leq_s \mathcal{F}_2^{\sigma_2}$ if and only if there is a $k \in \mathbb{N}$ such that for all $\mathsf{kb}_1 \in \mathcal{F}_1$ with $\sigma_1(\mathsf{kb}_1) \in \sigma_1(\mathcal{F}_1) \cap \sigma_2(\mathcal{F}_2)$, there is a $\mathsf{kb}_2 \in \mathcal{F}_2$ with $\sigma_1(\mathsf{kb}_1) = \sigma_2(\mathsf{kb}_2)$ and $\|\mathsf{kb}_2\|_2 \leq \|\mathsf{kb}_1\|_1^k$.

Intuitively, $\mathcal{F}_1^{\sigma_1} \leq_s \mathcal{F}_2^{\sigma_2}$ means that $\mathcal{F}_2$ under σ_2 is at least as succinct as $\mathcal{F}_1$ under σ_1. Put another way, for $\mathcal{F}_1^{\sigma_1} \leq_s \mathcal{F}_2^{\sigma_2}$ to hold, any knowledge base from $\mathcal{F}_1$ with an equivalent counterpart in $\mathcal{F}_2$ must have an equivalent counterpart *that is at most polynomially larger*. Note that succinctness talks only about those model sets that both can express, so it is most meaningful when comparing languages that are equally expressive, that is, whenever

3. For a set $X \subseteq 2^A$ we can simply define $\varphi_X = \bigvee_{M \in X} \varphi_M$ with $\varphi_M = \bigwedge_{a \in M} a \wedge \bigwedge_{a \in A \setminus M} \neg a$ and clearly $\mathit{mod}(\varphi_X) = X$.

$\sigma_1(\mathcal{F}_1) = \sigma_2(\mathcal{F}_2)$. As usual, we define $\mathcal{F}_1 <_s \mathcal{F}_2$ iff $\mathcal{F}_1 \leq_s \mathcal{F}_2$ and $\mathcal{F}_2 \not\leq_s \mathcal{F}_1$, and $\mathcal{F}_1 \cong_s \mathcal{F}_2$ iff $\mathcal{F}_1 \leq_s \mathcal{F}_2$ and $\mathcal{F}_2 \leq_s \mathcal{F}_1$. The relation $\leq_s$ is reflexive, but not necessarily antisymmetric or transitive.

The final general definition is about formalisms polynomially expressing languages. Here, we already make use of the previously introduced bijection between interpretations and binary words and use the term "languages" to synonymously refer to both.

Definition 3. A formalism $\mathcal{F}$ can polynomially express a language $L = \bigcup_{n\geq 0} L_n$ under semantics $\sigma : \mathcal{F} \to 2^{2^A}$ if and only if there is a $k \in \mathbb{N}$ such that for each positive $n \in \mathbb{N}$ there is a knowledge base $\mathsf{kb}_n \in \mathcal{F}$ of that formalism such that $\sigma(\mathsf{kb}_n) = L_n$ and $\|\mathsf{kb}_n\| \in O(n^k)$.

We next introduce some specific object-level languages that we will use. First of all, the language Parity contains all odd-element subsets of the vocabulary. Formally, for $A_n = \{a_1, \ldots, a_n\}$ with $n \geq 1$ we have

$$\textsc{Parity}_n = \{M \subseteq A_n \mid \exists m \in \mathbb{N} : |M| = 2m + 1\}$$

As explained before, then $\textsc{Parity} = \bigcup_{n\in\mathbb{N}, n\geq 1} \textsc{Parity}_n$. It is a textbook result that Parity is expressible by polynomial-size propositional formulas (Jukna, 2012); for example, we can define $\Phi_1^{\textsc{Parity}}(a_1) = a_1$ and for $n \geq 2$ set

$$\begin{aligned}\Phi_n^{\textsc{Parity}}(a_1, \ldots, a_n) = {} & (\Phi_{n_\downarrow}^{\textsc{Parity}}(a_1, \ldots, a_{n_\downarrow}) \land \neg\Phi_{n_\uparrow}^{\textsc{Parity}}(a_{n_\downarrow+1}, \ldots, a_n)) \lor \\ & (\neg\Phi_{n_\downarrow}^{\textsc{Parity}}(a_1, \ldots, a_{n_\downarrow}) \land \Phi_{n_\uparrow}^{\textsc{Parity}}(a_{n_\downarrow+1}, \ldots, a_n))\end{aligned}$$

with $n_\downarrow = \left\lfloor \frac{n}{2} \right\rfloor$ and $n_\uparrow = \left\lceil \frac{n}{2} \right\rceil$. (This construction yields a formula of logarithmic depth and therefore polynomial size.) It is also a textbook result (although not nearly as easy to see) that Parity cannot be expressed by depth-bounded polynomial-size circuits, that is, $\textsc{Parity} \notin \mathsf{AC}^0$ (Jukna, 2012).

As another important class, threshold languages are defined for $n, k \in \mathbb{N}$ with $n \geq 1$ and $k \leq n$:

$$\textsc{Threshold}_{n,k} = \{M \subseteq A_n \mid k \leq |M|\}$$

That is, $\textsc{Threshold}_{n,k}$ contains all interpretations over n atoms where at least k atoms are true. The special case $k = \left\lceil \frac{n}{2} \right\rceil$ leads to the majority languages,

$$\textsc{Majority}_n = \textsc{Threshold}_{n,\left\lceil \frac{n}{2} \right\rceil}$$

that contain all interpretations where at least half of the atoms in the vocabulary are true.

We next introduce the particular knowledge representation languages we study in this paper. All will make use of a vocabulary A; the results of the paper are all considered parametric in such a given vocabulary.

2.1 Logic Programs

For a vocabulary A we define $not\,A = \{not\,a \mid a \in A\}$ and accordingly the set of literals over A as $A^{\pm} = A \cup not\,A$. A *normal logic program rule* over A is then of the form $a \leftarrow B$ where $a \in A$ and $B \subseteq A^{\pm}$. The set B is called the *body* of the rule, we abbreviate $B^+ = B \cap A$ and

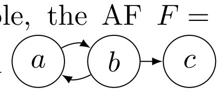
$B^- = \{a \in A \mid \mathit{not}\, a \in B\}$. A *logic program (LP)* P over A is a set of logic program rules over A. An interpretation $M \subseteq A$ *satisfies* the body B of a rule $a \leftarrow B \in P$ iff $B^+ \subseteq M$ and $B^- \cap M = \emptyset$. M is a *supported model* for P iff $M = \{a \in A \mid a \leftarrow B \in P, M \text{ satisfies } B\}$. For a logic program P we denote the set of its supported models by $su(P)$. The intuition behind this semantics is that the atoms that are true in a model are all and only those that have some kind of support.

However, this support might be cyclic self-support. For instance, the logic program $\{a \leftarrow \{a\}\}$ has two supported models, $\emptyset$ and $\{a\}$, where the latter is undesired in many application domains. As an alternative, Gelfond and Lifschitz (1988) proposed the stable model semantics, that does not allow self-support: A set $M \subseteq A$ is a *stable model* for P iff M is the $\subseteq$-least supported model of P^M, where P^M is obtained from P by (1) eliminating each rule whose body contains a literal $\mathit{not}\, a$ with $a \in M$, and (2) deleting all literals of the form $\mathit{not}\, a$ from the bodies of the remaining rules (Gelfond & Lifschitz, 1988). We write $st(P)$ for the set of stable models of P. It follows from the definition that $st(P)$ is a $\subseteq$-antichain: for all $M_1 \neq M_2 \in st(P)$ we have $M_1 \not\subseteq M_2$. As size measure we define $\|a \leftarrow B\| = |B| + 1$ for rules and $\|P\| = \sum_{r \in P} \|r\|$ for programs.

As an example, consider the vocabulary $A = \{a, b, c\}$ and over it the logic program $P = \{a \leftarrow \{b\}, b \leftarrow \{a\}, c \leftarrow \{\mathit{not}\, a\}\}$. We find $su(P) = \{\{c\}, \{a, b\}\}$ and $st(P) = \{\{c\}\}$.

2.2 Argumentation Frameworks

Dung (1995) introduced argumentation frameworks as pairs $F = (A, R)$ where A is a set of (abstract) arguments and $R \subseteq A \times A$ a relation of attack between the arguments. The purpose of semantics for argumentation frameworks is to determine sets of arguments (called *extensions*) which are acceptable according to various standards. For a given extension $S \subseteq A$, the arguments in S are considered to be accepted, those that are attacked by some argument in S are considered to be rejected, and all others are neither, their status is undecided. We will only be interested in so-called *stable* extensions, sets S of arguments that do not attack each other and attack all arguments not in the set. For stable extensions, each argument is either accepted or rejected by definition, thus the semantics is two-valued. More formally, a set $S \subseteq A$ of arguments is *conflict-free* iff there are no $a, b \in S$ with $(a, b) \in R$. A set S is a *stable extension* for (A, R) iff it is conflict-free and for all $a \in A \setminus S$ there is an argument $b \in S$ with $(b, a) \in R$. For an AF F, we denote the set of its stable extensions by $st(F)$. Again, it follows from the definition of a stable extension that the set $st(F)$ is always a $\subseteq$-antichain. The size of an argumentation framework $F = (A, R)$ is $\|F\| = |A| + |R|$.

For example, the AF $F = (\{a, b, c\}, \{(a, b), (b, a), (b, c)\})$ can be visualised using the directed graph  and has the set of stable extensions $st(F) = \{\{a, c\}, \{b\}\}$.

2.3 Abstract Dialectical Frameworks

An *abstract dialectical framework* is a tuple $D = (A, L, C)$ where A is a set of statements (representing positions one can take or not take in a debate), $L \subseteq A \times A$ is a set of links (representing dependencies between the positions), $C = \{C_a\}_{a \in A}$ is a collection of total functions $C_a : 2^{par(a)} \to \{\mathbf{t}, \mathbf{f}\}$, one for each statement $a \in A$. The function C_a is called *acceptance condition of* a and expresses whether a can be accepted, given the acceptance status of its parents $par(a)$. In this paper, we represent each C_a by a propositional formula φ_a over $par(a)$. As mentioned earlier, propositional formulas are built using negation $\neg$, conjunction $\wedge$ and disjunction $\vee$; connectives for material implication $\to$, logical equivalence $\leftrightarrow$ and exclusive disjunction $\not\leftrightarrow$ are regarded as abbreviations. To specify an acceptance condition, then, we take $C_a(M \cap par(a)) = \mathbf{t}$ to hold iff M is a model for φ_a, $M \models \varphi_a$.

Brewka and Woltran (2010) introduced a useful subclass of ADFs: an ADF $D = (A, L, C)$ is *bipolar* iff all links in L are supporting or attacking (or both). A link $(b, a) \in L$ is *supporting in* D iff for all $M \subseteq par(a)$, we have that $C_a(M) = \mathbf{t}$ implies $C_a(M \cup \{b\}) = \mathbf{t}$. Symmetrically, a link $(b, a) \in L$ is *attacking in* D iff for all $M \subseteq par(a)$, we have that $C_a(M \cup \{b\}) = \mathbf{t}$ implies $C_a(M) = \mathbf{t}$. If a link (b, a) is both supporting and attacking then b has no influence on a, the link is redundant (but does not violate bipolarity). We will sometimes use this circumstance when searching for ADFs; there we simply assume that $L = A \times A$, then links that are actually not needed can be expressed by acceptance conditions that make them redundant.

There are numerous semantics for ADFs; we will only be interested in two of them, (supported) models and stable models. A set $M \subseteq A$ is a *model of* D iff for all $a \in A$ we find that $a \in M$ iff $C_a(M) = \mathbf{t}$. The definition of stable models is inspired by logic programming and slightly more complicated (Brewka et al., 2013). Define an operator by[4]

$$\begin{aligned} \Gamma_D(X, Y) &= (ac(X, Y), re(X, Y)) \text{ for } X, Y \subseteq A, \text{ where} \\ ac(X, Y) &= \{a \in A \mid \forall Z \subseteq A : X \subseteq Z \subseteq A \setminus Y \Rightarrow C_a(Z) = \mathbf{t}\} \\ re(X, Y) &= \{a \in A \mid \forall Z \subseteq A : X \subseteq Z \subseteq A \setminus Y \Rightarrow C_a(Z) = \mathbf{f}\} \end{aligned}$$

The intuition behind the operator is as follows: A pair (X, Y) represents a partial interpretation of the set of statements where those in X are accepted (true), those in Y are rejected (false), and those in $A \setminus (X \cup Y)$ are neither. The operator checks for each statement a whether all total interpretations that can possibly arise from (X, Y) agree on their truth value for the acceptance condition of a. That is, if a has to be accepted no matter how the statements in $A \setminus (X \cup Y)$ are interpreted, then $a \in acc(X, Y)$. The set $rej(X, Y)$ is defined symmetrically, so the pair $(acc(X, Y), rej(X, Y))$ constitutes a refinement of (X, Y).

For $M \subseteq A$, the reduced ADF $D^M = (M, L^M, C^M)$ is defined by $L^M = L \cap M \times M$ and for each $a \in M$ setting $\varphi_a^M = \varphi_a[b/\bot : b \notin M]$, that is, replacing all $b \notin M$ by false in the acceptance formula of a. A model M for D is a *stable model* of D iff the least fixpoint of the operator Γ_{D^M} is given by $(M, \emptyset)$. As usual, $su(D)$ and $st(D)$ denote the respective model sets; while ADF models can be $\subseteq$-related, ADF stable models cannot. The size of an ADF D over A is given by $\|D\| = \sum_{a \in A} \|\varphi_a\|$; the size $\|\varphi\|$ of a formula φ is the number of its nodes.

As an example ADF D, consider vocabulary $A = \{a, b, c\}$ and the acceptance formulas $\varphi_a = c$, $\varphi_b = c$, and $\varphi_c = a \leftrightarrow b$. While D has a single supported model, $su(D) = \{\{a, b, c\}\}$, we find $st(D) = \emptyset$ since the atoms in the model support each other circularly.

2.4 Translations Between the Formalisms

We will review all known translations between the mentioned formalisms.

4. This operator is closely related to the ultimate approximation operators of Denecker, Marek, and Truszczyński (2004), as we observed earlier (Strass, 2013).

2.4.1 From AFs to BADFs

Brewka and Woltran (2010) showed how to translate AFs into ADFs: For an AF $F = (A, R)$, define the ADF associated to F as $D_F = (A, R, C)$ with $C = \{\varphi_a\}_{a \in A}$ and $\varphi_a = \bigwedge_{(b,a) \in R} \neg b$ for $a \in A$. Clearly, the resulting ADF is bipolar: parents are always attacking. Brewka and Woltran proved that this translation is faithful for the AF stable extension and ADF model semantics (Proposition 1). Brewka et al. (2013) later proved the same for the AF stable extension and ADF stable model semantics (Theorem 4). It is easy to see that the translation can be computed in polynomial time and induces at most a linear blowup.

2.4.2 From ADFs to PL

Brewka and Woltran (2010) also showed that ADFs under supported model semantics can be faithfully translated into propositional logic: when acceptance conditions of statements $a \in A$ are represented by propositional formulas φ_a, then the supported models of an ADF D over A are given by the classical propositional models of the formula set $\Phi_D = \{a \leftrightarrow \varphi_a \mid a \in A\}$.

2.4.3 From AFs to PL

In combination, the previous two translations yield a polynomial and faithful translation chain from AFs into propositional logic: $\Phi_{(A,R)} = \left\{a \leftrightarrow \left(\bigwedge_{(b,a) \in R} \neg b\right) \;\middle|\; a \in A\right\}$.

2.4.4 From ADFs to LPs

In earlier work (Strass, 2013), we showed that ADFs can be faithfully translated into normal logic programs. For an ADF $D = (A, L, C)$, its standard LP is

$$P_D = \{a \leftarrow (M \cup \mathit{not}\,(\mathit{par}(a) \setminus M)) \mid a \in A, C_a(M) = \mathbf{t}\}$$

It follows from Lemma 3.14 of Strass (2013) that this translation preserves the supported model semantics. The translation is size-preserving for the acceptance condition representation of Strass (2013) via characteristic models; when representing acceptance conditions via propositional formulas, this cannot be guaranteed as we will show later.[5]

2.4.5 From AFs to LPs

The translation chain from AFs to ADFs to LPs is compact, and faithful for AF stable semantics and LP stable semantics (Osorio, Zepeda, Nieves, & Cortés, 2005), and AF stable semantics and LP supported semantics (Strass, 2013). It is size-preserving since the single rule for each atom contains all attackers once: $P_{(A,R)} = \{a \leftarrow \{\mathit{not}\, b \mid (b,a) \in R\} \mid a \in A\}$.

5. Already for complexity reasons, we cannot expect that this translation is also faithful for the stable semantics. And indeed, the ADF $D = (\{a\}, \{(a,a)\}, \{\varphi_a = a \vee \neg a\})$ has a stable model $\{a\}$ while its standard logic program $P(D) = \{a \leftarrow \{a\}, a \leftarrow \{\mathit{not}\, a\}\}$ has no stable model. However, it holds that $\mathit{st}(P(D)) \subseteq \mathit{st}(D)$ (Denecker et al., 2004; Strass, 2013).

2.4.6 From LPs to PL

It is well-known that logic programs under supported model semantics can be translated to propositional logic (Clark, 1978). A logic program P becomes the propositional theory Φ_P,

$$\Phi_P = \{a \leftrightarrow \varphi_a \mid a \in A\} \quad \text{where} \quad \varphi_a = \bigvee_{a \leftarrow B \in P} \left(\bigwedge_{b \in B^+} b \wedge \bigwedge_{b \in B^-} \neg b \right) \quad \text{for } a \in A.$$

For the stable model semantics, additional formulas have to be added, but the extended translation works all the same (Lin & Zhao, 2004).

2.4.7 From LPs to ADFs

The Clark completion of a normal logic program directly yields an equivalent ADF over the same signature (Brewka & Woltran, 2010). Clearly the translation is computable in polynomial time and the blowup (with respect to the original logic program) is at most linear. The resulting translation is faithful for the supported model semantics, which follows from Lemma 3.16 of Strass (2013).

2.5 Representing Bipolar Boolean Functions

While bipolarity has hitherto predominantly been defined and used in the context of ADFs (Brewka & Woltran, 2010), it is easy to define the concept for Boolean functions in general. Let A be a set of atoms and $f : 2^A \to \{\mathbf{t}, \mathbf{f}\}$ be a Boolean function. An atom $a \in A$ is *supporting* iff for all $M \subseteq A$, $f(M) = \mathbf{t}$ implies $f(M \cup \{a\}) = \mathbf{t}$; we then write $a \in sup(f)$. An atom $a \in A$ is *attacking* iff for all $M \subseteq A$, $f(M) = \mathbf{f}$ implies $f(M \cup \{a\}) = \mathbf{f}$; we then write $a \in att(f)$. A Boolean function $f : 2^A \to \{\mathbf{t}, \mathbf{f}\}$ is *semantically bipolar* iff each $a \in A$ is supporting or attacking or both. Throughout the paper, we will sometimes take a Boolean function to be given by an interpretation set and then say that the set is bipolar.

We will now define bipolar propositional formulas for representing bipolar ADFs. This is important not only for our study, but also since (for three-valued semantics), bipolarity is the key to BADFs' low complexity in comparison to general ADFs (Strass & Wallner, 2015). Up to now, we usually assumed that to specify a bipolar ADF, in addition to statements, links and acceptance conditions, the user specifies for each link whether it is supporting or attacking (Strass & Wallner, 2015). Here we introduce an arguably simpler way, where support and attack is represented in the syntax of the propositional formula encoding the acceptance function.

Formally, the *polarity* of an atom $a \in A$ in a formula is determined by the number of negations on the path from the root of the formula tree to the atom. The polarity is *positive* if the number is even and *negative* if the number is odd.

Definition 4. A propositional formula φ over A is *syntactically bipolar* if and only if no atom $a \in A$ occurs both positively and negatively in φ.

Recall that we only use formulas over the basis $\{\wedge, \vee, \neg\}$ and thus there are no "hidden" negations, e.g. from material implication. For formulas in negation normal form (that is, where negation is only applied to atomic formulas), the polarities of the atoms can be read off the formula directly.

We will now address the question how to represent bipolar Boolean functions. Clearly all Boolean functions can be represented by propositional formulas; we modify this construction later and thus reproduce it here: for a Boolean function $f : 2^A \to \{\mathbf{t}, \mathbf{f}\}$, its associated formula is

$$\varphi_f = \bigvee_{M \subseteq A, f(M)=\mathbf{t}} \varphi_M \quad \text{with} \quad \varphi_M = \bigwedge_{a \in M} a \wedge \bigwedge_{a \in A \setminus M} \neg a \tag{1}$$

That is, each φ_M has exactly one model M, and φ_f enumerates those models.

So in particular, all bipolar Boolean functions can be represented by propositional formulas as well. However, this only guarantees us the existence of such representations but gives us no way to actually obtain them. Our first fundamental result shows how we can construct a syntactically bipolar propositional formula from a given semantically bipolar Boolean function. The converse is straightforward, and thus the two notions of bipolarity are closely related. For a formula φ, its associated Boolean function f_φ returns $\mathbf{t}$ if and only if it gets as input a model of φ.

Theorem 1. *Let A be a set of atoms.*

1. *For each syntactically bipolar formula φ over A, its Boolean function f_φ is semantically bipolar.*
2. *For each semantically bipolar Boolean function $f : 2^A \to \{\mathbf{t}, \mathbf{f}\}$, a syntactically bipolar formula ψ_f with $f_{\psi_f} = f$ is given by*

$$\psi_f = \bigvee_{\substack{M \subseteq A, \\ f(M)=\mathbf{t}}} \psi_M \quad \textit{with} \quad \psi_M = \bigwedge_{\substack{a \in M, \\ a \notin att(f)}} a \wedge \bigwedge_{\substack{a \in A \setminus M, \\ a \notin sup(f)}} \neg a \tag{2}$$

Proof. 1. Obvious: every atom occurring only positively is supporting, every atom occurring only negatively is attacking.

2. Let $f : 2^A \to \{\mathbf{t}, \mathbf{f}\}$ be semantically bipolar. Note first that by (2), for any $M \subseteq A$ we have $\models \varphi_M \to \psi_M$. It is easy to see that ψ_f is syntactically bipolar: Since f is semantically bipolar, each $a \in A$ is: (1) attacking and not supporting, then it occurs only negatively in ψ_f; or (2) supporting and not attacking, then it occurs only positively in ψ_f; or (3) supporting and attacking, then it does not occur in ψ_f. It remains to show that $f_{\psi_f} = f$; we show $\models \varphi_f \leftrightarrow \psi_f$.

$\models \varphi_f \to \psi_f$: Let $v : A \to \{\mathbf{t}, \mathbf{f}\}$ with $v(\varphi_f) = \mathbf{t}$. Then there is an $M \subseteq A$ such that $f(M) = \mathbf{t}$ and $v(\varphi_M) = \mathbf{t}$. (Clearly $v = v_M$.) By $\models \varphi_M \to \psi_M$ we get $v(\psi_M) = \mathbf{t}$ and thus $v(\psi_f) = \mathbf{t}$.

$\models \psi_f \to \varphi_f$: For each model v of ψ_f, there is an $M \subseteq A$ with $f(M) = \mathbf{t}$ such that $v(\psi_M) = \mathbf{t}$. To show that each model of ψ_f is a model of φ_f, we show that for all $M \subseteq A$ with $f(M) = \mathbf{t}$, each model v of ψ_M is a model of φ_f. Let $|A| = n$. Then each φ_M contains exactly n literals. For the corresponding ψ_M there is a $k \in \mathbb{N}$ with $0 \leq k \leq n$ such that ψ_M contains exactly $n - k$ literals. For two interpretations $v_1 : A \to \{\mathbf{t}, \mathbf{f}\}$ and $v_2 : A \to \{\mathbf{t}, \mathbf{f}\}$, define the *difference* between them as $\delta(v_1, v_2) = \{a \in A \mid v_1(a) \neq v_2(a)\}$. (Note that for $|A| = n$ we always have $|\delta(v_1, v_2)| \leq n$.) We will use induction on k to show the following: for each $M \subseteq A$ with $f(M) = \mathbf{t}$, each $v : A \to \{\mathbf{t}, \mathbf{f}\}$ with $v(\psi_M) = \mathbf{t}$ and $|\delta(v, v_M)| = k$ we find that $v(\varphi_f) = \mathbf{t}$. This covers all models v of ψ_f (since $|\delta(v, v_M)| \leq |A|$) and thus establishes the claim.

$k = 0$: $\delta(v, v_M) = \emptyset$ implies $v = v_M$ whence $v(\varphi_f) = v_M(\varphi_f) = v_M(\varphi_M) = \mathbf{t}$ by definition of φ_M and φ_f.

$k \rightsquigarrow k+1$: Let $M \subseteq A$ with $f(M) = \mathbf{t}$, and $v : A \to \{\mathbf{t}, \mathbf{f}\}$ with $v(\psi_M) = \mathbf{t}$ and $|\delta(v, v_M)| = k + 1$. Since $k + 1 > 0$, there is some $a \in \delta(v, v_M)$, that is, an $a \in A$ with $v(a) \neq v_M(a)$.

(a) a is supporting and not attacking. Then necessarily $v(a) = \mathbf{t}$. (If $v(a) = \mathbf{f}$, then $v_M(a) \neq v(a)$ implies $v_M(a) = \mathbf{t}$, that is, $a \in M$ whence $\{\psi_M\} \models a$ and $v(\psi_M) = \mathbf{f}$, contradiction.) Define the interpretation $w : A \to \{\mathbf{t}, \mathbf{f}\}$ such that $w(a) = \mathbf{f}$ and $w(c) = v(c)$ for $c \in A \setminus \{a\}$. Clearly $\delta(v, w) = \{a\}$ and $|\delta(w, v_M)| = k$. Hence the induction hypothesis applies to w and $w(\varphi_f) = \mathbf{t}$. Now $w(a) = \mathbf{f}$, $v(a) = \mathbf{t}$ and $w(\varphi_f) = \mathbf{t}$. Since a is supporting, also $v(\varphi_f) = \mathbf{t}$.

(b) a is attacking and not supporting. Symmetric to the opposite case above.

(c) a is both supporting and attacking. Define interpretation $w : A \to \{\mathbf{t}, \mathbf{f}\}$ such that $w(a) = v_M(a)$ and $w(c) = v(c)$ for $c \in A \setminus \{a\}$. It follows that $|\delta(w, v_M)| = k$, whence the induction hypothesis applies to w and $w(\varphi_f) = \mathbf{t}$. Since a is both supporting and attacking (thus redundant), we get that $v(\varphi_f) = w(\varphi_f) = \mathbf{t}$. □

This result paves the way for analysing the succinctness of bipolar ADFs, since now we have a quite natural way of representing them.

3. Relative Expressiveness

We now analyse and compare the relative expressiveness of argumentation frameworks (AFs), (bipolar) abstract dialectical frameworks ((B)ADFs), normal logic programs (LPs) and propositional logic (PL). We first look at the different families of semantics – supported and stable models – in isolation and afterwards combine the results for the two semantics. For the formalisms $\mathcal{F} \in \{\mathrm{ADF}, \mathrm{LP}\}$ that have both supported and stable semantics, we will indicate the semantics σ via a superscript as in Definition 1. For AFs we only consider the stable semantics, as this is (to date) the only semantics for AFs where all interpretations are guaranteed to map all arguments to either true (accepted) or false (rejected, i.e. attacked by an accepted argument). For propositional logic PL we consider the usual model semantics.

With the syntactic translations we reviewed in the previous section, we currently have the following expressiveness relationships. For the supported semantics,

$$\mathrm{AF} \leq_e \mathrm{BADF}^{su} \leq_e \mathrm{ADF}^{su} \cong_e \mathrm{LP}^{su} \leq_e \mathrm{PL}$$

and for the stable semantics,

$$\text{AF} \leq_e \text{LP}^{st} <_e \text{PL and AF} \leq_e \text{BADF}^{st} \leq_e \text{ADF}^{st} <_e \text{PL}$$

Note that $\text{LP}^{st} <_e \text{PL}$ and $\text{ADF}^{st} <_e \text{PL}$ hold since sets of stable models have an antichain property, in contrast to model sets of propositional logic.

For the succinctness relation, we have

$$\text{AF} \leq_s \text{BADF}^{su} \leq_s \text{ADF}^{su} \leq_s \text{PL and LP}^{su} \leq_s \text{ADF}^{su}$$

3.1 Supported Semantics

As depicted above, we know that expressiveness from AFs to propositional logic does not decrease. However, it is not yet clear if any of the relationships is strict. In what follows we will show that two of them are strict, working our way top-down from most to least expressive.

3.1.1 ADF vs. PL

We first show that ADFs can realise any set of models by showing how a given propositional formula can be used to construct an equivalent ADF of linear size.[6]

Theorem 2. *PL* $\leq_e$ *ADF*^{su} *and PL* $\leq_s$ *ADF*^{su}.

Proof. Let ψ be a propositional formula over vocabulary A. Define the ADF D_ψ over A by setting, for all $a \in A$,

$$\varphi_a = a \leftrightarrow \psi = (a \land \psi) \lor (\neg a \land \neg \psi)$$

Thus $\|\varphi_a\| \in O(\|\psi\|)$, whence $\|D_\psi\| \in O(|A| \cdot \|\psi\|)$. It remains to show $su(D_\psi) = mod(\psi)$. Recall that for any ADF D over A, $su(D) = mod(\Phi_D)$ for $\Phi_D = \bigwedge_{a \in A} (a \leftrightarrow \varphi_a)$. Applying the definition of φ_a in D_ψ yields

$$\Phi_{D_\psi} = \bigwedge_{a \in A} (a \leftrightarrow (a \leftrightarrow \psi))$$

Now for any $a \in A$, the formula $(a \leftrightarrow (a \leftrightarrow \psi))$ is equivalent to ψ. (The proof is by case distinction on a.) Thus Φ_{D_ψ} is equivalent to $\bigwedge_{a \in A} \psi$, that is, to ψ, and it follows that $su(D_\psi) = mod(\Phi_{D_\psi}) = mod(\psi)$. □

For example, consider the vocabulary $A = \{a, b\}$ and the propositional formula $\psi = a \land b$. The canonical construction above yields ADF D_ψ with acceptance formulas $\varphi_a = a \leftrightarrow (a \land b)$ and $\varphi_b = b \leftrightarrow (a \land b)$. Now we have:

$$\varphi_a = a \leftrightarrow (a \land b) = (a \to (a \land b)) \land ((a \land b) \to a) \equiv \neg a \lor (a \land b) \equiv \neg a \lor b$$

Intuitively, $\varphi_a = \neg a \lor b$ expresses that a cannot be false, and is true if b is true. By a symmetrical argument, the acceptance formula of b is equivalent to $\neg b \lor a$. It is readily checked that $su(D_\psi) = \{\{a, b\}\}$ as desired. Since we know from Section 2.4.2 that the converse translation is also possible ($\text{ADF}^{su} \leq_s \text{PL}$), we get the following.

Corollary 3. *PL* $\cong_s$ *ADF*^{su}

6. If we consider the vocabulary A to be part of the input, the size increase is quadratic.

When the acceptance conditions are written as propositional formulas, the construction to realise $X \subseteq 2^A$ in the proof of Theorem 2 defines a space-efficient equivalent of

$$\varphi_a = \bigvee_{M \in X, a \in M} \varphi_M \vee \bigvee_{M \subseteq A, M \notin X, a \notin M} \varphi_M$$

as acceptance formula of a, where φ_M is as in Footnote 3.

3.1.2 ADF vs. LP

Since ADFs under supported semantics can be faithfully translated into logic programs, which can be likewise further translated to propositional logic, we have the following.

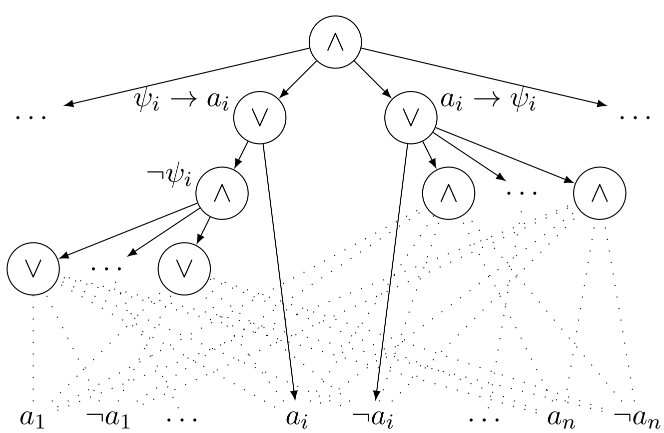
Corollary 4. *$ADF^{su} \cong_e LP^{su} \cong_e PL$*

However, this does not extend to the succinctness relation, as logic programs stipulate a particular syntactic form that is essentially a fixed-depth circuit. More specifically, it is easy to see that any language that is polynomially expressible by normal logic programs under supported semantics is in AC^0. For the stable semantics of so-called canonical logic programs, this has recently been shown by Shen and Zhao (2014) (Proposition 2.1). The case we are interested in (supported semantics) works similarly, but we still present the proof for completeness. The main technical result towards proving that is a lemma showing how to turn a logic program into an equivalent Boolean circuit of a fixed depth.

Lemma 5. *For every normal logic program P, there exists a circuit C_P over the basis $\{\neg, \wedge, \vee\}$ such that:*

1. *C_P accepts all and only the supported models of P,*
2. *the size of C_P is linear the size of P,*
3. *C_P has depth 4.*

Proof. Let $A = \{a_1, \dots, a_n\}$ be the vocabulary of P, and its Clark completion be $\Phi_P = \{a_i \leftrightarrow \psi_i \mid a_i \in A\}$ where the ψ_i are DNFs over literals from A. Clearly the circuit for Φ_P must compute $C_P = \bigwedge_{a_i \in A}(a_i \leftrightarrow \psi_i)$ where $a_i \leftrightarrow \psi_i$ can be replaced by $(\neg a_i \vee \psi_i) \wedge (a_i \vee \neg \psi_i)$ with $\neg\psi_i$ a CNF over literals from A. The construction can be depicted as follows, where the inner layers are shown for one i only, and dotted lines represent potential edges.



Now (1) follows since $su(P) = mod(\Phi_P)$ and C_P accepts all and only the models of Φ_P. For (2), if P contains $m = |P|$ rules, then $m \leq \|P\|$ and the total number of inner gates is bounded by $n(2m + 3) \leq n(2 \cdot \|P\| + 3)$. (3) is clear. $\square$

While the statement of Lemma 5 is actually much stronger and gives a *constant* upper bound of the resulting circuit depth for arbitrarily-sized logic programs, it readily follows that the set of polynomially logic-program expressible languages is a subset of the languages expressible by alternating Boolean circuits with unbounded fan-in and constant depth.

Proposition 6. *If L is polynomially expressible by normal logic programs under supported semantics, then $L \in \mathsf{AC}^0$.*

It follows immediately that normal logic programs cannot polynomially express the language Parity.[7] This is the supported-semantics counterpart of Theorem 3.1 in (Shen & Zhao, 2014).

Corollary 7. Parity *has no polynomial size normal logic program representation.*

Proof. By Proposition 6 and Parity $\notin \mathsf{AC}^0$ (Jukna, 2012). $\square$

It follows that propositional logic is strictly more succinct than normal logic programs under supported semantics.

Corollary 8. *$PL \not\leq_s LP^{su}$ and thus $LP^{su} <_s PL$.*

From our considerations since Theorem 2, it follows that if ψ has a "small" conjunctive normal form (a conjunction of clauses) and disjunctive normal form (disjunction of monomials) representation, then there is also a "small" normal logic program representation for $mod(\psi)$.

3.1.3 ADF vs. BADF

It is quite obvious that the canonical ADF constructed in Theorem 2 is not bipolar, since a as well as every atom mentioned by ψ occurs both positively and negatively in φ_a. This raises the question whether the construction can be adapted to *bipolar* ADFs.

It turns out that the subclass of bipolar ADFs is strictly less expressive. Towards the proof of this result we start out with a new concept: that of the conjugate of a model set with respect to an atom. This concept will be used to characterise ADF realisability and precisely captures the "if-and-only-if part" of ADFs' supported model semantics: From the translation of an ADF D into propositional logic (cf. Section 2.4.2) we can see that the result is basically a conjunction of equivalences: $\phi_D = \bigwedge_{a \in A}(a \leftrightarrow \varphi_a)$. While the conjunction part will be captured by set intersection, the conjugate will capture the equivalence part.

Definition 5. Let A be a vocabulary, $X \subseteq 2^A$ and $a \in A$. The *a-conjugate of X* is the set

$$\langle a \rangle(X) = \{M \subseteq A \mid M \in X, a \in M\} \cup \{M \subseteq A \mid M \notin X, a \notin M\}$$

7. Logic programs under supported models are universally expressive, so they *can* express Parity, just not in polynomial size.

Alternatively, we could write $\langle a\rangle(X) = \{M \subseteq A \mid M \in X \leftrightarrow a \in M\}$. Intuitively, $\langle a\rangle(X)$ contains all interpretations M where containment of a in M coincides exactly with containment of M in X. Formulated in terms of propositional formulas, if X is the model set of formula φ over A, then $\langle a\rangle(X)$ is the model set of formula $a \leftrightarrow \varphi$. Note that the vocabulary A is implicit in the conjugate function.

Example 1. Consider the vocabulary $A_2 = \{a, b\}$. The functions $\langle a\rangle(\cdot)$ and $\langle b\rangle(\cdot)$ operate on the set $2^{2^{A_2}}$ of interpretation sets over A_2 and are shown in Table 1.

φ	$\langle a\rangle(\varphi)$	$\langle b\rangle(\varphi)$
$\bot$	$\neg a$	$\neg b$
$\neg a \wedge \neg b$	$\neg a \wedge b$	$a \wedge \neg b$
$a \wedge \neg b$	$\neg a \vee \neg b$	$\neg a \wedge \neg b$
$\neg a \wedge b$	$\neg a \wedge \neg b$	$\neg a \vee \neg b$
$a \wedge b$	$a \rightarrow b$	$b \rightarrow a$
a	$\top$	$a \leftrightarrow b$
b	$a \leftrightarrow b$	$\top$
$\neg a$	$\bot$	$a \not\leftrightarrow b$
$\neg b$	$a \not\leftrightarrow b$	$\bot$
$a \not\leftrightarrow b$	$\neg b$	$\neg a$
$a \leftrightarrow b$	b	a
$a \vee b$	$b \rightarrow a$	$a \rightarrow b$
$\neg a \vee \neg b$	$a \wedge \neg b$	$\neg a \wedge b$
$a \rightarrow b$	$a \wedge b$	$a \vee b$
$b \rightarrow a$	$a \vee b$	$a \wedge b$
$\top$	a	b

Table 1: Conjugation functions for $A_2 = \{a, b\}$. Interpretation sets are represented using formulas over A_2, and connective "$\not\leftrightarrow$" denotes exclusive disjunction XOR.

For two-valued ADF semantics, this conjugation function plays an essential semantical role, since it provides the "bridge" between models of the acceptance functions and models of the ADF. But it is also interesting in itself: We first show some properties of the conjugation function associated to an atom, since some of them will be used in the proof later on. First of all, it is an involution, that is, its own inverse (and thus in particular a bijection). Next, it is compatible with the complement operation (logical negation on the formula level). Finally, it also preserves the evenness of the cardinality of the input set.

Proposition 9. *Let A be a vocabulary, $X \subseteq 2^A$ and $a \in A$.*

1. $\langle a\rangle(\langle a\rangle(X)) = X$. *(involution)*
2. $2^A \setminus \langle a\rangle(X) = \langle a\rangle\big(2^A \setminus X\big)$. *(compatible with negation)*
3. *$|X|$ is even iff $|\langle a\rangle(X)|$ is even.* *(preserves evenness)*

Proof. Let $|A| = n$, $X \subseteq 2^A$ and $a \in A$.

1. Let $M \subseteq A$. We have

$$\begin{aligned} M \in \langle a\rangle(\langle a\rangle(X)) &\text{ iff } M \in \langle a\rangle(X) \leftrightarrow a \in M \\ &\text{ iff } (M \in X \leftrightarrow a \in M) \leftrightarrow a \in M \\ &\text{ iff } M \in X \leftrightarrow (a \in M \leftrightarrow a \in M) \\ &\text{ iff } M \in X \end{aligned}$$

2. Denote

$$\begin{aligned} S_{\in,\in} &= \{M \subseteq A \mid M \in X, a \in M\} \\ S_{\in,\notin} &= \{M \subseteq A \mid M \in X, a \notin M\} \\ S_{\notin,\in} &= \{M \subseteq A \mid M \notin X, a \in M\} \\ S_{\notin,\notin} &= \{M \subseteq A \mid M \notin X, a \notin M\} \end{aligned}$$

and observe that

$$\begin{aligned} 2^A &= S_{\in,\in} \uplus S_{\in,\notin} \uplus S_{\notin,\in} \uplus S_{\notin,\notin} \\ X &= S_{\in,\in} \uplus S_{\in,\notin} \\ \langle a\rangle(X) &= S_{\in,\in} \uplus S_{\notin,\notin} \end{aligned}$$

where $\uplus$ denotes disjoint union. Now

$$\begin{aligned} 2^A \setminus \langle a\rangle(X) &= 2^A \setminus \left(S_{\in,\in} \uplus S_{\notin,\notin}\right) \\ &= S_{\in,\notin} \uplus S_{\notin,\in} \\ &= \{M \subseteq A \mid M \in X, a \notin M\} \uplus \{M \subseteq A \mid M \notin X, a \in M\} \\ &= \left\{M \subseteq A \;\middle|\; M \notin 2^A \setminus X, a \notin M\right\} \uplus \left\{M \subseteq A \;\middle|\; M \in 2^A \setminus X, a \in M\right\} \\ &= \langle a\rangle\left(2^A \setminus X\right) \end{aligned}$$

3. We show that $|X| + |\langle a\rangle(X)|$ is even. Firstly,

$$S_{\in,\notin} \uplus S_{\notin,\notin} = \{M \subseteq A \mid a \notin M\} = 2^{A \setminus \{a\}}$$

whence $\left|S_{\in,\notin}\right| + \left|S_{\notin,\notin}\right| = 2^{n-1}$. Thus

$$\begin{aligned} |X| + |\langle a\rangle(X)| &= \left(|S_{\in,\in}| + \left|S_{\in,\notin}\right|\right) + \left(|S_{\in,\in}| + \left|S_{\notin,\notin}\right|\right) \\ &= 2 \cdot |S_{\in,\in}| + \left|S_{\in,\notin}\right| + \left|S_{\notin,\notin}\right| \\ &= 2 \cdot |S_{\in,\in}| + 2^{n-1} \end{aligned}$$

is even. □

For our current purpose of characterising the expressiveness of bipolar ADFs, we now use the concept of conjugation to make ADF realisability for the model semantics slightly more accessible. We show that each ADF realisation of a model set X over an n-element vocabulary A can equivalently be characterised by an n-tuple $(Y_1, \ldots, Y_n)$ of supersets of X whose intersection is exactly X. The crux of the proof of this result is how the acceptance conditions of the realising ADF and the Y_i are related through the conjugation function.

Proposition 10. *Let $A = \{a_1, \ldots, a_n\}$ be a vocabulary and $X \subseteq 2^A$ be a set of interpretations. Denote an ADF over A by the sequence $(\varphi_1, \ldots, \varphi_n)$ of its acceptance formulas (for each $i \in \{1, \ldots, n\}$, formula φ_i over A is the acceptance formula of a_i), and further define*

$$C_X = \{(mod(\varphi_1), \ldots, mod(\varphi_n)) \mid su(\varphi_1, \ldots, \varphi_n) = X\}$$
$$Y_X = \left\{(Y_1, \ldots, Y_n) \;\middle|\; Y_1, \ldots, Y_n \subseteq 2^A, \left(\bigcap_{i=1}^{n} Y_i\right) = X\right\}$$

The sets C_X and Y_X are in one-to-one correspondence; in particular $|C_X| = |Y_X|$.

Proof. We provide a bijection between C_X and Y_X. Consider the function

$$f : \left(2^{2^A}\right)^n \to \left(2^{2^A}\right)^n \quad \text{with} \quad (B_1, \ldots, B_n) \mapsto (\langle a_1\rangle(B_1), \ldots, \langle a_n\rangle(B_n))$$

which is an involution by Proposition 9. Using the results of Section 2.4.2, we get that

$$\begin{aligned}
(mod(\varphi_1), \ldots, mod(\varphi_n)) \in C_X \text{ iff } & su(\varphi_1, \ldots, \varphi_n) = X \\
\text{iff } & mod\left(\bigwedge_{1\le i\le n} (a_i \leftrightarrow \varphi_i)\right) = X \\
\text{iff } & \bigcap_{1\le i\le n} mod(a_i \leftrightarrow \varphi_i) = X \\
\text{iff } & \bigcap_{1\le i\le n} \langle a_i\rangle(mod(\varphi_i)) = X \\
\text{iff } & (\langle a_1\rangle(mod(\varphi_1)), \ldots, \langle a_n\rangle(mod(\varphi_n))) \in Y_X \\
\text{iff } & f(mod(\varphi_1), \ldots, mod(\varphi_n)) \in Y_X
\end{aligned}$$

Thus $f(C_X) = Y_X$ whence $f(Y_X) = f(f(C_X)) = C_X$ and $f|_{C_X} : C_X \to Y_X$ is bijective. $\square$

This one-to-one correspondence is important since we will later analyse the precise number of realisations of given model sets. Furthermore, this result shows the role of the conjugation function for characterising two-valued model realisability for general ADFs. We can now adapt this characterisation result to the case of bipolar ADFs. More precisely, we give several necessary and sufficient conditions when a given model set is bipolarly realisable. With this characterisation in hand, we can later show that a specific interpretation set fails the necessary conditions and thus cannot be the model set of any BADF. Below, we denote the set of all supersets of a set X of interpretation sets over A by $X^{\uparrow} = \left\{Y \subseteq 2^A \;\middle|\; X \subseteq Y\right\}$.

Proposition 11. *Let $A = \{a_1, \ldots, a_n\}$ be a vocabulary and $X \subseteq 2^A$ be a set of interpretations. The following are equivalent:*

1. *X is bipolarly realisable.*
2. *there exist $Y_1, \ldots, Y_n \in X^{\uparrow}$ such that:*
   (a) *$(\bigcap_{i=1}^{n} Y_i) = X$, and*

(b) *for each* $1 \le i \le n$, *the set* $\langle a_i \rangle (Y_i)$ *is bipolar.*

3. *there exist* $Y_1, \ldots, Y_n \in X^{\uparrow}$ *such that*

   (a) $(\bigcap_{i=1}^{n} Y_i) = X$, *and*

   (b) *for each* $1 \le i, j \le n$, *at least one of:*

   - *for all* $M \subseteq A$, $(M \in Y_i \leftrightarrow a_i \in M) \to (M \cup \{a_j\} \in Y_i \leftrightarrow a_i \in M \cup \{a_j\})$; *or*
   - *for all* $N \subseteq A$, $(N \in Y_i \not\leftrightarrow a_i \in N) \to (N \cup \{a_j\} \in Y_i \not\leftrightarrow a_i \in N \cup \{a_j\})$.

Proof. (1) $\Rightarrow$ (2): If X is bipolarly realisable, then there exists a bipolar ADF $D = (A, L, C)$ with $su(D) = X$. In particular, there exist bipolar Boolean functions $C_1, \ldots, C_n$ such that $M \in X$ if and only if for all $1 \le i \le n$ we find $a_i \in M$ iff $C_i(M) = \mathbf{t}$. For each $1 \le i \le n$ define $Y_i = \langle a_i \rangle (C_i)$. By assumption, $\langle a_i \rangle (Y_i) = \langle a_i \rangle (\langle a_i \rangle (C_i)) = C_i$ is bipolar; furthermore $(\bigcap_{i=1}^{n} Y_i) = X$ follows from the above.

(2) $\Rightarrow$ (3): Let $i \in \{1, \ldots, n\}$ and assume that $\langle a_i \rangle (Y_i)$ is bipolar. This means that for all $a_j \in A$, we find that a_j is supporting or attacking (or both) in $\langle a_i \rangle (Y_i)$. Now a_j is supporting in $\langle a_j \rangle (Y_i)$ iff for all $M \subseteq A$ we find:

$$M \in \langle a_i \rangle (Y_i) \to M \cup \{a_j\} \in \langle a_i \rangle (Y_i)\,, \text{ that is,}$$
$$(M \in Y_i \leftrightarrow a_i \in M) \to (M \cup \{a_j\} \in Y_i \leftrightarrow a_i \in M \cup \{a_j\})$$

Similarly, a_j is attacking in $\langle a_i \rangle (Y_i)$ iff for all $N \subseteq A$ we find:

$$N \notin \langle a_i \rangle (Y_i) \to N \cup \{a_j\} \notin \langle a_i \rangle (Y_i)\,, \text{ that is,}$$
$$\neg(N \in Y_i \leftrightarrow a_i \in N) \to \neg(N \cup \{a_j\} \in Y_i \leftrightarrow a_i \in N \cup \{a_j\})$$

Thus for all $a_j \in A$, we find that at least one of the following:

- for all $M \subseteq A$, $(M \in Y_i \leftrightarrow a_i \in M) \to (M \cup \{a_j\} \in Y_i \leftrightarrow a_i \in M \cup \{a_j\})$; or
- for all $N \subseteq A$, $(N \in Y_i \not\leftrightarrow a_i \in N) \to (N \cup \{a_j\} \in Y_i \not\leftrightarrow a_i \in N \cup \{a_j\})$.

(3) $\Rightarrow$ (1): We construct an ADF $D = (A, L, C)$ as follows: for each $i \in \{1, \ldots, n\}$ we define $C_i = \langle a_i \rangle (Y_i)$ and finally set $L = A \times A$. Each C_i is bipolar by the equivalences established in the previous proof item, and $su(D) = X$ follows from the fact that $\langle a_i \rangle (C_i) = \langle a_i \rangle (\langle a_i \rangle (Y_i)) = Y_i$ and the presumption $(\bigcap_{i=1}^{n} Y_i) = X$. □

We now apply this characterisation result to show that there is an interpretation set over three atoms that cannot be realised by bipolar ADFs under the model semantics. This is the smallest example in terms of the number of atoms (actually, one of the two smallest examples) – all interpretation sets over a binary vocabulary are bipolarly realisable.

Proposition 12. *For vocabulary* $A_3 = \{1, 2, 3\}$, *there is no bipolar ADF that realises* $X = \textsc{Even}_3 = \{\emptyset, \{1, 2\}, \{1, 3\}, \{2, 3\}\}$.

Proof. Assume to the contrary that X is bipolarly realisable. Then there exist $Y_1, Y_2, Y_3 \in X^{\uparrow}$ from Proposition 11. There are $2^{|2^A|-|X|} = 2^{8-4} = 2^4 = 16$ candidates for each Y_i, that is, every Y_i must be of the form $X \uplus Z$ with

$$Z \subseteq \{\{1\}, \{2\}, \{3\}, \{1,2,3\}\} = 2^A \setminus X$$

For eleven out of those sixteen model set candidates for each Y_i, the set $\langle i\rangle(Y_i)$ is not bipolar. To show that a model set $\langle i\rangle(Y_i)$ is not bipolar, we provide a statement $j \in A_3$ that is neither supporting nor attacking; we say that such a statement is *dependent*.

1. For $Y_1 = X$, we get $\langle 1\rangle(Y_1) = \{\{1,2\}, \{1,3\}, \{2\}, \{3\}\}$, which is not bipolar since statement 2 is dependent: If 2 was supporting, then $\{3\} \in \langle 1\rangle(Y_1)$ would imply $\{2,3\} \in \langle 1\rangle(Y_1)$; if 2 was attacking, then $\emptyset \notin \langle 1\rangle(Y_1)$ would imply $\{2\} \notin \langle 1\rangle(Y_1)$. For the remaining cases, the justifications for a specific statement being dependent are equally easy to read off the model set; for brevity we just indicate the statements.
2. For $Y_1 = X \cup \{\{1\}\}$, we get $\langle 1\rangle(Y_1) = \{\{1,2\}, \{1,3\}, \{1\}, \{2\}, \{3\}\}$, which is not bipolar since statement 2 is dependent.
3. For $Y_1 = X \cup \{\{2\}\}$, we get $\langle 1\rangle(Y_1) = \{\{1,2\}, \{1,3\}, \{3\}\}$, which is not bipolar since statement 2 is dependent.
4. The case $Y_1 = X \cup \{\{3\}\}$ is symmetric to the previous one: we get the model set $\langle 1\rangle(Y_1) = \{\{1,2\}, \{1,3\}, \{2\}\}$, which is not bipolar since statement 3 is dependent.
5. For $Y_1 = X \cup \{\{1,2,3\}\}$, we get $\langle 1\rangle(Y_1) = \{\{1,2,3\}, \{1,2\}, \{1,3\}, \{2\}, \{3\}\}$, which is not bipolar since statement 2 is dependent.
6. For $Y_1 = X \cup \{\{1\}, \{2\}\}$, we get $\langle 1\rangle(Y_1) = \{\{1,2\}, \{1,3\}, \{1\}, \{3\}\}$, which is not bipolar since statement 3 is dependent.
7. The case $Y_1 = X \cup \{\{1\}, \{3\}\}$ is again symmetric to the previous one.
8. For $Y_1 = X \cup \{\{2\}, \{3\}\}$, we get $\langle 1\rangle(Y_1) = \{\{1,2\}, \{1,3\}\}$, which is not bipolar since statement 2 is dependent.
9. For $Y_1 = X \cup \{\{1\}, \{1,2,3\}\}$, we get $\langle 1\rangle(Y_1) = \{\{1,2,3\}, \{1,2\}, \{1,3\}, \{1\}, \{2\}, \{3\}\}$, which is not bipolar since statement 2 is dependent.
10. For $Y_1 = X \cup \{\{2\}, \{1,2,3\}\}$, we get $\langle 1\rangle(Y_1) = \{\{1,2,3\}, \{1,2\}, \{1,3\}, \{3\}\}$, which is not bipolar since statement 2 is dependent.
11. $Y_1 = X \cup \{\{3\}, \{1,2,3\}\}$ is again symmetric to the previous case.

There remains a set C of five candidates (due to symmetry they are the same for each i):

$$\begin{aligned} C = \{&X \uplus \{\{1\}, \{2\}, \{3\}\}, \\ &X \uplus \{\{1\}, \{2\}, \{1,2,3\}\}, \\ &X \uplus \{\{1\}, \{3\}, \{1,2,3\}\}, \\ &X \uplus \{\{2\}, \{3\}, \{1,2,3\}\}, \\ &X \uplus \{\{1\}, \{2\}, \{3\}, \{1,2,3\}\}\} \end{aligned}$$

Basically, the candidates are those where at least three out of the four interpretations in $D = \{\{1\}, \{2\}, \{3\}, \{1,2,3\}\}$ are contained in addition to those already in X. Now clearly by the assumption that the Y_i realise X we have $Y_1, Y_2, Y_3 \in C$. But then there is some $M \in D$ with $M \in Y_i$ for all $1 \leq i \leq 3$ and thus $M \in \left(\bigcap_{i=1}^{3} Y_i\right) = X$. However, $D \cap X = \emptyset$. Contradiction. Thus such Y_i do not exist and X is not bipolarly realisable. $\square$

As the only other interpretation set over A_3 that is not bipolarly realisable, we found the complement of Even_3 above, the Parity language over three atoms.

Proposition 13. *For vocabulary $A_3 = \{1,2,3\}$, there is no bipolar ADF that realises* $\text{Parity}_3 = \{\{1\}, \{2\}, \{3\}, \{1,2,3\}\}$.

Together with the straightforward statement of fact that Even_3 can be realised by a non-bipolar ADF, Proposition 12 leads to the next result.

Theorem 14. *$BADF^{su} <_e ADF^{su}$*

Proof. Model set Even_3 from Proposition 12 is realisable under model semantics by ADF D_{Even_3} with acceptance conditions

$$\varphi_1 = (2 \not\leftrightarrow 3), \quad \varphi_2 = (1 \not\leftrightarrow 3), \quad \varphi_3 = (1 \not\leftrightarrow 2)$$

However, there is no bipolar ADF realising Even_3, as is witnessed by Proposition 12. $\square$

Another consequence of our characterisation of two-valued model realisability in Proposition 10 is that we can get a precise number of distinct realisations of a given model set. This is significant in that it further illustrates the rather intricate difficulty underlying bipolar non-realisability: we cannot necessarily use the model set Even_3 above to determine a *single* reason for bipolar non-realisability, that is, a *single* link (b, a) that is neither supporting nor attacking in *all* realisations. Rather, the culprit(s) might be different in each realisation, and to show bipolar non-realisability, we have to prove that *for all* realisations, there necessarily *exists some* reason for non-bipolarity. And the number of different ADF realisations of a given model set X can be considerable.[8]

Proposition 15. *Let A be a vocabulary with $|A| = n$, and $X \subseteq 2^A$ an interpretation set with $\left|2^A \setminus X\right| = m$. The number of distinct ADFs D with $su(D) = X$ is*

$$r(n, m) = (2^n - 1)^m$$

Proof. According to Proposition 10, each realisation of X can be characterised by a tuple $(Y_1, \ldots, Y_n) \in \left(X^\uparrow\right)^n$ with $X = \bigcap_{i=1}^{n} Y_i$. Since $\left|X^\uparrow\right| = 2^m$, there are $(2^m)^n$ such tuples. However, towards $r(n,m)$, this wrongly counts all tuples $(Y_1, \ldots, Y_n)$ with $(\bigcap_{i=1}^{n} Y_i) \supsetneq X$, that is, $|(\bigcap_{i=1}^{n} Y_i) \setminus X| > 0$ (at least once); it remains to subtract them. For any $i \in \{1, \ldots, n\}$, we can overestimate the number of tuples $(Y_1, \ldots, Y_n) \in \left(X^\uparrow\right)^n$ such that $|(\bigcap_{i=1}^{n} Y_i) \setminus X| \geq i$ by the expression

$$q(n, m, i) = \binom{m}{i} \left(2^{m-i}\right)^n \tag{3}$$

8. When counting ADFs over A, we do not take into account different link relations, but take $L = A \times A$ and only count different acceptance functions, through which redundant links can be modelled.

This is seen as follows: Let $I \subseteq \left(2^A \setminus X\right)$ be a fixed i-element set. (Intuitively, the interpretation-set $X \cup I$ contains exactly i interpretations too many.) There are $\binom{m}{i}$ such sets. For each such I, we have $\left|I^{\uparrow}\right| = 2^{m-i}$. Thus there are $\left(2^{m-i}\right)^n$ possible ways to choose n elements (the $Y_1, \ldots, Y_n$) out of $I^{\uparrow}$. No matter how the Y_j are chosen, their intersection contains I and thus has at least i elements too many. However, all sets that have at least $i+1$ elements too many are counted twice and have to be subtracted. If we subtract $q(n,m,i+1)$, then we have not counted the sets that have at least $i+2$ elements too many and have to add $q(n,m,i+2)$, etc. Hence by the inclusion-exclusion principle, the number of tuples $(Y_1, \ldots, Y_n) \in \left(X^{\uparrow}\right)^n$ with $\bigcap_{i=1}^n Y_i = X$ is given by

$$\begin{aligned}
r(n,m) &= q(n,m,0) - q(n,m,1) + q(n,m,2) - \ldots \pm q(n,m,m) \\
&= \sum_{i=0}^{m} (-1)^i q(n,m,i) \\
&= \sum_{i=0}^{m} (-1)^i \binom{m}{i} \left(2^{m-i}\right)^n && \text{(by (3) above)} \\
&= \sum_{i=0}^{m} \binom{m}{i} \left(2^n\right)^{m-i} (-1)^i && \text{(reordering factors)} \\
&= \left(2^n - 1\right)^m && \text{(binomial theorem)} \quad \square
\end{aligned}$$

So the main contributing factor is the number m of interpretations that are excluded from the desired model set X. For Proposition 12, for instance, there are $(2^3-1)^4 = 7^4 = 2401$ ADFs with the model set Even_3. According to Theorem 14, none of them is bipolar. Obviously, the maximal number of realisations is achieved by $X = \emptyset$ whence $r(n, 2^n) = (2^n - 1)^{2^n}$. On the other hand, the model set $X = 2^A$ has exactly one realisation, $r(n,0) = 1$. Note that the number of (syntactically distinct) realisations for the other universally expressive formalisms, logic programs and propositional logic, is unbounded in general since we can add an arbitrary number of tautologies.

We finally show a reduction of the problem of bipolar realisability to propositional satisfiability. This approaches the problem from another angle (a possible implementation deciding bipolar realisability using a SAT solver), and provides the proof of Theorem 3 by Strass (2015b), which was not contained in that work.

For a given vocabulary A and set $X \subseteq 2^A$ be a set of interpretations, it is our aim to construct a propositional formula ϕ_X that is satisfiable if and only if X is bipolarly realisable. The propositional signature we use is the following: For each $a \in A$ and $M \subseteq A$, there is a propositional variable p_a^M that expresses whether $C_a(M) = \mathbf{t}$. This allows to encode all possible acceptance conditions for the statements in A. To enforce bipolarity, we use additional variables to model supporting and attacking links: for all $a, b \in A$, there is a variable $p_{sup}^{a,b}$ saying that a supports b, and a variable $p_{att}^{a,b}$ saying that a attacks b. So the vocabulary of ϕ_X is given by

$$P = \left\{ p_a^M, p_{sup}^{a,b}, p_{att}^{a,b} \;\middle|\; M \subseteq A, a \in A, b \in A \right\}$$

To guarantee the desired set of models, we constrain the acceptance conditions as dictated by X: For any desired set M and statement a, the containment of a in M must correspond exactly to whether $C_a(M) = \mathbf{t}$; this is encoded in $\phi_X^{\in}$. Conversely, for any undesired set M and statement a, there must not be any such correspondence, which $\phi_X^{\notin}$ expresses. To enforce bipolarity, we state that each link must be supporting or attacking. To model the meaning of support and attack, we encode all ground instances of their definitions.

Definition 6. Let A be a vocabulary and $X \subseteq 2^A$ be a set of interpretations. Define the following propositional formulas:

$$\begin{aligned}
\phi_X^{\mathrm{BADF}} &= \phi_X^{\in} \wedge \phi_X^{\notin} \wedge \phi_{bipolar} \\
\phi_X^{\in} &= \bigwedge_{M \in X} \left(\bigwedge_{a \in M} p_a^M \wedge \bigwedge_{a \in A \setminus M} \neg p_a^M \right) \\
\phi_X^{\notin} &= \bigwedge_{M \subseteq A, M \notin X} \left(\bigvee_{a \in M} \neg p_a^M \vee \bigvee_{a \in A \setminus M} p_a^M \right) \\
\phi_{bipolar} &= \bigwedge_{a,b \in A} \left(\left(p_{sup}^{a,b} \vee p_{att}^{a,b} \right) \wedge \phi_{sup}^{a,b} \wedge \phi_{att}^{a,b} \right) \\
\phi_{sup}^{a,b} &= p_{sup}^{a,b} \to \bigwedge_{M \subseteq A} \left(p_b^M \to p_b^{M \cup \{a\}} \right) && (a, b \in A) \\
\phi_{att}^{a,b} &= p_{att}^{a,b} \to \bigwedge_{M \subseteq A} \left(p_b^{M \cup \{a\}} \to p_b^M \right) && (a, b \in A)
\end{aligned}$$

The corresponding result shows the reduction to be correct.

Theorem 16. *Let A be a vocabulary and $X \subseteq 2^A$ be a set of interpretations. X is bipolarly realisable if and only if ϕ_X^{BADF} is satisfiable.*

Proof. "if": Let $I \subseteq P$ be a model for ϕ_X. For each $a \in A$, we define an acceptance condition as follows: for $M \subseteq A$, set $C_a(M) = \mathbf{t}$ iff $p_a^M \in I$. It is easy to see that $\phi_{bipolar}$ guarantees that these acceptance conditions are all bipolar. The ADF is now given by $D_X^{su} = (A, A \times A, C)$. It remains to show that any $M \subseteq A$ is a model of D_X^{su} if and only if $M \in X$.

"if": Let $M \in X$. We have to show that M is a model of D_X^{su}. Consider any $a \in A$.

1. $a \in M$. Since I is a model of $\phi_X^{\in}$, we have $p_a^M \in I$ and thus by definition $C_a(M) = \mathbf{t}$.
2. $a \in A \setminus M$. Since I is a model of $\phi_X^{\in}$, we have $p_a^M \notin I$ and thus by definition $C_a(M) = \mathbf{f}$.

"only if": Let $M \notin X$. Since I is a model of $\phi_X^{\notin}$, there is an $a \in M$ such that $C_a(M) = \mathbf{f}$ or an $a \notin M$ such that $C_a(M) = \mathbf{t}$. In any case, M is not a model of D_X^{su}.

"only if": Let D be a bipolar ADF with $su(D) = X$. We use D to define a model I for ϕ_X. First, for $M \subseteq A$ and $a \in A$, set $p_a^M \in I$ iff $C_a(M) = \mathbf{t}$. Since D is bipolar, each link is supporting or attacking and for all $a, b \in A$ we can find a valuation for $p_{sup}^{a,b}$ and $p_{att}^{a,b}$. It remains to show that I is a model for ϕ_X.

1. I is a model for $\phi_X^{\in}$: Since D realises X, each $M \in X$ is a model of D and thus for all $a \in A$ we have $C_a(M) = \mathbf{t}$ iff $a \in M$.
2. I is a model for $\phi_X^{\notin}$: Since D realises X, each $M \subseteq A$ with $M \notin X$ is not a model of D. Thus for each such M, there is an $a \in A$ witnessing that M is not a model of D: (1) $a \in M$ and $C_a(M) = \mathbf{f}$, or (2) $a \notin M$ and $C_a(M) = \mathbf{t}$.
3. I is a model for $\phi_{bipolar}$: straightforward since D is bipolar by assumption. □

Remarkably, the decision procedure does not only give an answer, but in the case of a positive answer we can read off the BADF realisation from the satisfying evaluation of the constructed formula. We illustrate the construction with an example seen earlier.

Example 2. Consider $A_3 = \{1, 2, 3\}$ and the model set $\text{Even}_3 = \{\emptyset, \{1, 2\}, \{1, 3\}, \{2, 3\}\}$. The construction of Theorem 16 yields these formulas:

$$\begin{aligned}\phi^{\in}_{\text{Even}_3} = {} & \neg p_1^{\emptyset} \wedge \neg p_2^{\emptyset} \wedge \neg p_3^{\emptyset} \wedge \\ & p_1^{\{1,2\}} \wedge p_2^{\{1,2\}} \wedge \neg p_3^{\{1,2\}} \wedge \\ & p_1^{\{1,3\}} \wedge \neg p_2^{\{1,3\}} \wedge p_3^{\{1,3\}} \wedge \\ & \neg p_1^{\{2,3\}} \wedge p_2^{\{2,3\}} \wedge p_3^{\{2,3\}}\end{aligned}
\qquad
\begin{aligned}\phi^{\notin}_{\text{Even}_3} = {} & (\neg p_1^{\{1\}} \vee p_2^{\{1\}} \vee p_3^{\{1\}}) \wedge \\ & (p_1^{\{2\}} \vee \neg p_2^{\{2\}} \vee p_3^{\{2\}}) \wedge \\ & (p_1^{\{3\}} \vee p_2^{\{3\}} \vee \neg p_3^{\{3\}}) \wedge \\ & (\neg p_1^{\{1,2,3\}} \vee \neg p_2^{\{1,2,3\}} \vee \neg p_3^{\{1,2,3\}})\end{aligned}$$

The remaining formulas about bipolarity are independent of Even_3, we do not show them here. We have implemented the translation of the proof of Theorem 16 and used the solver clasp (Gebser, Kaminski, Kaufmann, Ostrowski, Schaub, & Schneider, 2011) to verify that ϕ_{Even_3} is unsatisfiable.

3.1.4 BADF vs. LP

Earlier, we used the language Parity to show that propositional logic is (and thus by $\text{PL} \cong_s \text{ADF}^{su}$ general ADFs are) exponentially more succinct than normal logic programs (under supported models). However, for bipolar ADFs, by Proposition 13 there is no BADF D over $A_3 = \{1, 2, 3\}$ with model set $su(D) = \text{Parity}_3 = \{\{1\}, \{2\}, \{3\}, \{1, 2, 3\}\}$, that is, BADFs cannot even express Parity. Fortunately, the Majority language does the trick in this case.

Theorem 17. *$\text{BADF}^{su} \not\leq_s \text{LP}^{su}$*

Proof. We show that the language Majority can be polynomially expressed by BADF^{su}, but not by LP^{su}. The latter fact follows from $\text{Majority} \notin \mathsf{AC}^0$ (Jukna, 2012) and Proposition 6. We show the first part by constructing a series of BADFs D_n over $A_n = \{a_1, \ldots, a_n\}$ ($n \in \mathbb{N}$, $n \geq 1$) such that $su(D_n) = \text{Majority}_n$. We use results of (Friedman, 1986; Boppana, 1986), who show that for all positive $n \in \mathbb{N}$ and $k \leq n$, the language $\text{Threshold}_{n,k}$ has negation-free propositional formulas $\Phi^{\text{Threshold}}_{n,k}$ of polynomial size s, where we use the bound of Boppana, $s \in O\big(k^{4.27} n \log n\big)$. Define D_1 by $\varphi_{a_1} = \top$, and for $n \geq 2$ set $k = \lceil \frac{n}{2} \rceil$ and for $1 \leq i \leq n$,

$$\varphi_{a_i} = a_i \vee \neg\Phi^{\text{Threshold}}_{n-1,k}(a_1, \ldots, a_{i-1}, a_{i+1}, \ldots, a_n)$$

Intuitively, the formula φ_{a_i} checks whether the remaining variables could achieve a majority without a_i. If so, then a_i can be set arbitrarily; otherwise, a_i must be set to true. Clearly the Boolean function computed by φ_{a_i} is bipolar, since a_i is supporting and all other parents are attacking. For the size of D_n, we observe that

$$\|D_n\| \in O\big(n \left\|\Phi^{\textsc{Threshold}}_{n-1,k}\right\|\big)$$

whence the overall size is polynomial. It remains to show that $su(D_n) = \textsc{Majority}_n$.

"$\supseteq$": Let $M \in \textsc{Majority}_n$. We have to show $M \in su(D_n)$, that is, $a \in M$ iff $M \models \varphi_a$ for all $a \in A_n$. For $a \in M$, it is immediate that $M \models \varphi_a$, so let $a_j \notin M$ for some $j \in \{1, \ldots, n\}$. We have to show $M \not\models \varphi_{a_j}$. Since $M \in \textsc{Majority}_n$, we have $|M| = m$ for $k = \left\lceil \frac{n}{2} \right\rceil \leq m \leq n-1$ and $M \in \textsc{Threshold}_{n-1,k}$, that is, we have

$$M \models \Phi^{\textsc{Threshold}}_{n-1,k}(a_1, \ldots, a_{j-1}, a_{j+1}, \ldots, a_n)$$

Together with $M \not\models a_j$, it follows that $M \not\models \varphi_{a_j}$.

"$\subseteq$": Let $M \notin \textsc{Majority}_n$. Then $|M| = m$ for $0 \leq m < \left\lceil \frac{n}{2} \right\rceil = k$. In particular, there is some $a_j \in A_n \setminus M$. Now $m < k$ implies that there is no $N \in \textsc{Threshold}_{n-1,k}$ with $|N| = m = |M|$. Thus $M \not\models \Phi^{\textsc{Threshold}}_{n-1,k}(a_1, \ldots, a_{j-1}, a_{j+1}, \ldots, a_n)$ whence it follows that $M \models \varphi_{a_j}$. Together with $M \not\models a_j$ we conclude that $M \notin su(D_n)$. □

Since every BADF is an ADF of the same size, we get:

Corollary 18. *$ADF^{su} \not\leq_s LP^{su}$*

In combination with the translation from logic programs to ADFs (implying the relation $\mathrm{LP}^{su} \leq_s \mathrm{ADF}^{su}$), this means that also ADFs are strictly more succinct than logic programs.

Corollary 19. *$LP^{su} <_s ADF^{su}$*

3.1.5 BADF vs. AF

It is comparably easy to show that BADF models are strictly more expressive than AFs, since sets of supported models of bipolar ADFs do not have the antichain property.

Proposition 20. *$AF <_e BADF^{su}$*

Proof. Consider vocabulary $A = \{a\}$ and BADF $D = (A, \{(a,a)\}, \{\varphi_a\})$ with $\varphi_a = a$. It is straightforward to check that its model set is $su(D) = \{\emptyset, \{a\}\}$. Since model sets of AFs under stable extension semantics satisfy the antichain property, there is no equivalent AF over A. □

This yields the following overall relationships:

$$\mathrm{AF} <_e \mathrm{BADF}^{su} <_e \mathrm{ADF}^{su} \cong_e \mathrm{LP}^{su} \cong_e \mathrm{PL}$$

For a concise overview of relative succinctness, we present the results and open problems at a glance in Table 2 below.[9]

9. We remark that the three open problems in Table 2 are really only two: It is easy to show that ADFs and propositional logic behave equivalently in relation to bipolar ADFs, since they are equally expressive and equally succinct; that is, it holds that $\mathrm{ADF}^{su} \leq_s \mathrm{BADF}^{su}$ if and only if $\mathrm{PL} \leq_s \mathrm{BADF}^{su}$.

	BADF^{su}	ADF^{su}	LP^{su}	PL
BADF^{su}	$=$	$\leq_s$	$\not\leq_s$	$\leq_s$
ADF^{su}	?	$=$	$\not\leq_s$	$\cong_s$
LP^{su}	?	$<_s$	$=$	$<_s$
PL	?	$\cong_s$	$\not\leq_s$	$=$

Table 2: Relative succinctness results for (bipolar) ADFs under the model semantics, normal logic programs under the supported semantics, and classical propositional logic. An entry $\circ$ in row $\mathcal{F}_1$ and column $\mathcal{F}_2$ means $\mathcal{F}_1 \circ \mathcal{F}_2$.

3.2 Stable Semantics

As before, we recall the current state of knowledge:

$$\text{AF} \leq_e \text{BADF}^{st} \leq_e \text{ADF}^{st} <_e \text{PL and AF} \leq_e \text{LP}^{st} <_e \text{PL}$$

We first show that BADFs are strictly more expressive than AFs.

Proposition 21. *AF $<_e$ BADFst*

Proof. Consider the set $X_2 = \{\{a, b\}, \{a, c\}, \{b, c\}\}$ of desired models. Dunne et al. (2015) proved that X_2 is not realisable with stable AF semantics. However, the model set X_2 is realisable with BADF D_{X_2} under stable semantics:

$$\varphi_a = \neg b \vee \neg c, \qquad \varphi_b = \neg a \vee \neg c, \qquad \varphi_c = \neg a \vee \neg b$$

Let us exemplarily show that $M = \{a, b\}$ is a stable model (the other cases are completely symmetric): The reduct D^M is characterised by the two acceptance formulas $\varphi_a = \neg b \vee \neg\bot$ and $\varphi_b = \neg a \vee \neg\bot$. We then easily find that $\Gamma_{D^M}(\emptyset, \emptyset) = (M, \emptyset) = \Gamma_{D^M}(M, \emptyset)$. $\square$

Intuitively, the argument for AF non-realisability of X_2 is as follows: Since a and b occur in an extension together, there can be no attack between them. The same holds for the pairs a, c and b, c. But then the set $\{a, b, c\}$ is conflict-free and thus there must be a stable extension containing all three arguments, which is not allowed by X_2. The reason is AFs' restriction to individual attack, as set attack (also called joint or collective attack) suffices to realise X_2 as seen above.

The construction that we used in the proof above to realize X_2 comes from the work of Eiter, Fink, Pührer, Tompits, and Woltran (2013) in logic programming, and can be generalised to realise any non-empty model set satisfying the antichain property.

Definition 7. Let $X \subseteq 2^A$. Define the following BADF $D_X^{st} = (A, L, C)$ where C_a for $a \in A$ is given by

$$\varphi_a = \bigvee_{M \in X, a \in M} \left(\bigwedge_{b \in A \setminus M} \neg b \right)$$

and thus $L = \{(b, a) \mid M \in X, a \in M, b \in A \setminus M\}$.

The next result shows that the construction indeed works.

Theorem 22. *Let X with $\emptyset \neq X \subseteq 2^A$ be a $\subseteq$-antichain. We find that $st(D_X^{st}) = X$.*

Proof. Let $M \subseteq A$.

"$\subseteq$": Let $M \notin X$. We show that $M \notin su(D_X^{st}) \supseteq st(D_X^{st})$; we use a case distinction.

1. There is an $N \in X$ with $M \subsetneq N$. Then there is an $a \in N \setminus M$. Consider its acceptance formula φ_a. Since $a \in N$ and $N \in X$, the formula φ_a has a disjunct $\psi_{a,N} = \bigwedge_{b \in A \setminus N} \neg b$. Now $M \subseteq N$ implies $A \setminus N \subseteq A \setminus M$ and M is a model for $\psi_{a,N}$. Thus M is a model for φ_a although $a \notin M$, hence $M \notin su(D_X^{st})$.
2. For all $N \in X$, we have $M \not\subseteq N$. Then $X \neq \emptyset$ implies $M \neq \emptyset$, so let $a \in M$. For each $N \in X$ with $a \in N$, the acceptance formula φ_a contains a disjunct $\psi_{a,N} = \bigwedge_{b \in A \setminus N} \neg b$. By assumption, for each $N \in X$ there is a $b_N \in M \setminus N$. Clearly $b_N \in A \setminus N$ and b_N is evaluated to true by M. Hence for each $N \in X$ with $a \in N$, the disjunct $\psi_{a,N}$ is evaluated to false by M. Thus φ_a is false under M and $M \notin su(D_X^{st})$.

"$\supseteq$": Let $M \in X$. We first show that $M \in su(D_X^{st})$, that is: for all $a \in A$, we find $a \in M$ iff M is a model for φ_a.

1. Let $a \in M$. By construction, we have that φ_a in D_X^{st} contains a disjunct of the form $\psi_{a,M} = \bigwedge_{b \in A \setminus M} \neg b$. According to the interpretation M, all such $b \in A \setminus M$ are false and thus $\psi_{a,M}$ is true whence φ_a is true.
2. Let $a \in A \setminus M$ and consider its acceptance formula φ_a. Assume to the contrary that M is a model for φ_a. Then there is some $N \in X$ with $a \in N$ such that M is a model for $\psi_{a,N} = \bigwedge_{b \in A \setminus N} \neg b$, that is, $A \setminus N \subseteq A \setminus M$. Hence $M \subseteq N$; and, since $a \in N \setminus M$, even $M \subsetneq N$, whence X is not a $\subseteq$-antichain. Contradiction. Thus M is no model for φ_a.

Now consider the reduct D^M of D_X^{st} with respect to M. There, φ_a^M contains the disjunct $\psi_{a,M}^M = \psi_{a,M}[b/\bot : b \notin M]$ where all $b \in A \setminus M$ have been replaced by false, whence $\psi_{a,M}^M = \neg\bot \land \ldots \land \neg\bot$ and φ_a^M is equivalent to true. Thus each $a \in M$ is true in the least fixpoint of Γ_{D^M} and thus $M \in st(D_X^{st})$. □

The restriction to non-empty model sets is immaterial for relative expressiveness, since we can use the construction of Theorem 2 and the fact that $st(D) \subseteq su(D)$ for any ADF D to realize the empty model set. As the stable model semantics for ADFs and logic programs both have the antichain property, we get:

Corollary 23. *$\mathrm{ADF}^{st} \leq_e \mathrm{BADF}^{st}$ and $\mathrm{LP}^{st} \leq_e \mathrm{BADF}^{st}$*

This leads to the following overall relationships:

$$\mathrm{AF} <_e \mathrm{BADF}^{st} \cong_e \mathrm{ADF}^{st} \cong_e \mathrm{LP}^{st} <_e \mathrm{PL}$$

We remark that the antichain property provides a *characterisation* of realisability with the stable semantics; that is, a model set is stable-realisable iff it is a $\subseteq$-antichain.

3.3 Supported vs. Stable Semantics

Now we put the supported and stable pictures together. From the proof of Theorem 22, we can read off that for the canonical realisation D_X^{st} of an antichain X, the supported and stable semantics coincide, that is, $su(D_X^{st}) = st(D_X^{st}) = X$. With this observation, also bipolar ADFs under the supported semantics can realize any antichain, and we have this:

Proposition 24. $\mathit{BADF}^{st} \leq_e \mathit{BADF}^{su}$

As we have seen in Proposition 20, there are bipolar ADFs with supported-model sets that are not antichains. We get:

Corollary 25. $\mathit{BADF}^{st} <_e \mathit{BADF}^{su}$

This result allows us to close the last gap and put together the big picture on relative expressiveness in Figure 2 below.

$$\begin{array}{c} \mathrm{ADF}^{su} \cong_e \mathrm{LP}^{su} \cong_e \mathrm{PL} \\ | \\ \mathrm{BADF}^{su} \\ | \\ \mathrm{BADF}^{st} \cong_e \mathrm{ADF}^{st} \cong_e \mathrm{LP}^{st} \\ | \\ \mathrm{AF} \end{array}$$

Figure 2: The expressiveness hierarchy. Expressiveness strictly increases from bottom to top. $\mathcal{F}^\sigma$ denotes formalism $\mathcal{F}$ under semantics σ, where "su" is the supported and "st" the stable model semantics; formalisms are among AFs (argumentation frameworks), ADFs (abstract dialectical frameworks), BADFs (bipolar ADFs), LPs (normal logic programs) and PL (propositional logic).

4. Allowing Vocabulary Expansion

Up to here, we only considered *compact* realisations, that do not introduce new vocabulary elements. In this section, we allow the introduction of a small number of new atoms/arguments/statements. More precisely, *small* means the number is linear in the size of the source knowledge base (representing the model set that we wish to realize in a target language). For the purpose of realisability, the new vocabulary elements are projected out of the resulting models.

As it turns out, adding additional arguments already makes AFs universally expressive (under projection). More technically, we will now show that for each propositional formula φ over vocabulary A, there exists an AF F_φ over an expanded vocabulary $A \cup A_\varphi$ such that the models of φ and the stable extensions of F_φ correspond one-to-one. Roughly, this is possible since AFs can be regarded as a syntactic variant of classical propositional logic that has as its only connective the logical NOR "$\downarrow$" (Gabbay, 2011; Brewka et al., 2011). Using this connective, negation is expressed by $\neg\varphi = \varphi \downarrow \varphi$ and disjunction by

$\varphi \vee \psi = \neg(\varphi \downarrow \psi) = (\varphi \downarrow \psi) \downarrow (\varphi \downarrow \psi)$. These equivalences can be used to translate arbitrary propositional formulas (over $\neg, \wedge, \vee$) into the syntactical $\downarrow$-fragment; to guarantee that the size increase is at most linear, we introduce names a_ψ for subformulas ψ (Tseitin, 1968). The next definition combines all of these ideas.

Definition 8. Let φ be a formula using $\neg, \wedge, \vee$ over vocabulary A. Define the sets A_φ and R_φ inductively as follows:

$$
\begin{aligned}
A_\top &= \{a_\top\} & R_\top &= \emptyset \\
A_\bot &= \{a_\bot\} & R_\bot &= \{(a_\bot, a_\bot)\} \\
A_p &= \{p, a_{\neg p}\} \text{ for } p \in A & R_p &= \{(p, a_{\neg p}), (a_{\neg p}, p)\} \text{ for } p \in A \\
A_{\neg\xi} &= \{a_{\neg\xi}\} \cup A_\xi & R_{\neg\xi} &= \{(a_\xi, a_{\neg\xi})\} \cup R_\xi \\
A_{\zeta\wedge\xi} &= \{a_{\zeta\wedge\xi}, a_{\neg\zeta}, a_{\neg\xi}\} \cup A_{\neg\zeta} \cup A_{\neg\xi} & R_{\zeta\wedge\xi} &= \{(a_{\neg\zeta}, a_{\zeta\wedge\xi}), (a_{\neg\xi}, a_{\zeta\wedge\xi})\} \cup R_{\neg\zeta} \cup R_{\neg\xi} \\
A_{\zeta\vee\xi} &= \{a_{\zeta\vee\xi}, a_{\zeta\downarrow\xi}\} \cup A_\zeta \cup A_\xi & R_{\zeta\vee\xi} &= \{(a_{\zeta\downarrow\xi}, a_{\zeta\vee\xi}), (a_\zeta, a_{\zeta\downarrow\xi}), (a_\zeta, a_{\zeta\downarrow\xi})\} \cup R_\zeta \cup R_\xi
\end{aligned}
$$

The *AF associated to* φ is given by $F_\varphi = (A_\varphi \cup A_\bot, R_\varphi \cup \{(a_\varphi, a_\bot)\} \cup R_\bot)$.

The argument $a_\top$ is unattacked and thus part of every stable extension (is true in every interpretation); the argument $a_\bot$ attacks itself and thus cannot be part of any stable extension (is false in every interpretation). The mutually attacking arguments p and $a_{\neg p}$ for $p \in A$ serve to "guess" a valuation of A, while a_φ and $a_\bot$ guarantee that only (and all) valuations that are models of φ can lead to stable extensions of F_φ: intuitively, $a_\bot$ must be attacked, and the only candidate to do so is a_φ. The arguments and attacks for the Boolean connectives express their usual truth-theoretic semantics, as our first technical result for this translation shows.

Lemma 26. *Let φ be a formula over vocabulary A and F_φ its associated AF. For each stable extension M of F_φ and $a_\zeta, a_\xi \in A_\varphi$, we have:*

- *$a_{\neg\xi} \in M$ iff $a_\xi \notin M$;*
- *$a_{\zeta\wedge\xi} \in M$ iff both $a_\zeta \in M$ and $a_\xi \in M$;*
- *$a_{\zeta\vee\xi} \in M$ iff one of $a_\zeta \in M$ or $a_\xi \in M$;*
- *$a_{\zeta\downarrow\xi} \in M$ iff neither $a_\zeta \in M$ nor $a_\xi \in M$.*

Proof.
- By definition, the only attacker of an argument of the form $a_{\neg\xi}$ is the argument a_ξ. Thus $a_\xi \in M$ iff $a_{\neg\xi} \notin M$.
- The only attackers of $a_{\zeta\wedge\xi}$ are the arguments $a_{\neg\zeta}$ and $a_{\neg\xi}$. By the case above, we have $a_{\neg\zeta} \in M$ iff $a_\zeta \notin M$, and $a_{\neg\xi} \in M$ iff $a_\xi \notin M$. Consequently, $a_\zeta \in M$ and $a_\xi \in M$ iff $a_{\neg\zeta} \notin M$ and $a_{\neg\xi} \notin M$ iff $a_{\zeta\wedge\xi} \in M$.
- The only attacker of $a_{\zeta\vee\xi}$ is the argument $a_{\zeta\downarrow\xi}$. Similarly to the previous cases, we can show that $a_{\zeta\downarrow\xi} \in M$ iff $a_\zeta \notin M$ and $a_\xi \notin M$, and that $a_{\zeta\vee\xi} \in M$ iff $a_{\zeta\downarrow\xi} \notin M$. In combination, $a_{\zeta\vee\xi} \in M$ iff $a_\zeta \in M$ or $a_\xi \in M$.

- The only attackers of $a_{\zeta\downarrow\xi}$ are the arguments a_ζ and a_ξ. It directly follows that $a_{\zeta\downarrow\xi} \in M$ iff neither $a_\zeta \in M$ nor $a_\xi \in M$. $\square$

These correspondences can be used to show by induction that the newly introduced arguments capture the semantics of the formulas they encode (for all subformulas ψ of φ).

Lemma 27. *Let φ be a formula over A and F_φ its associated AF. For each stable extension M of F_φ and $a_\psi \in A_\varphi$, we have $a_\psi \in M$ iff $M \cap A$ is a model of ψ.*

Proof. Let M be a stable extension of F. We use structural induction on ψ.

$\psi = \top$: Trivial: $a_\top \in M$ since it has no attackers.

$\psi = \bot$: Trivial: $a_\bot \notin M$ since the set $\{a_\bot\}$ is not conflict-free.

$\psi = p \in A$: Trivial: $p \in M$ iff $M \models p$ by definition.

$\psi = \neg\xi$: $a_\psi \in M$ iff $a_{\neg\xi} \in M$ iff $a_\xi \notin M$ iff $M \not\models \xi$ iff $M \models \neg\xi$ iff $M \models \psi$.

$\psi = \zeta \wedge \xi$: $a_\psi \in M$ iff $a_{\zeta\wedge\xi} \in M$ iff $a_\zeta \in M$ and $a_\xi \in M$ iff $M \models \zeta$ and $M \models \xi$ iff $M \models \zeta \wedge \xi$ iff $M \models \psi$.

$\psi = \zeta \vee \xi$: $a_\psi \in M$ iff $a_{\zeta\vee\xi} \in M$ iff $a_\zeta \in M$ or $a_\xi \in M$ iff $M \models \zeta$ or $M \models \xi$ iff $M \models \zeta \vee \xi$ iff $M \models \psi$.

$\psi = \zeta \downarrow \xi$: $a_\psi \in M$ iff $a_{\zeta\downarrow\xi} \in M$ iff $a_\zeta \notin M$ and $a_\xi \notin M$ iff $M \not\models \zeta$ and $M \not\models \xi$ iff $M \models \zeta \downarrow \xi$ iff $M \models \psi$. $\square$

This lets us show the main result of this section, namely that the AF stable extension semantics is universally expressive under projection.

Theorem 28. *Let φ be a formula over vocabulary A and F_φ its associated AF.*

1. *For each model $M \subseteq A$ of φ, there exists a stable extension E of F_φ with $M = E \cap A$.*
2. *For each stable extension E of F_φ, the set $E \cap A$ is a model of φ.*

Proof. 1. Let $M \subseteq A$ be a model of φ. Define the set

$$E = \{a_\psi \mid a_\psi \in A_\varphi, M \models \psi\}$$

Observe that $M = E \cap A$. By presumption, $a_\varphi \in E$. It remains to show that E is a stable extension, that is, E is conflict-free and attacks all arguments $b \notin E$.

E is conflict-free: Assume to the contrary that there is an attack $r = (a, b) \in R_\varphi$ with $a, b \in E$. By definition, there are only these cases:

- a is arbitrary and $b = \bot$. But then by definition of E we get $M \models \bot$, contradiction.
- $r = (p, a_{\neg p})$ or $r = (a_{\neg p}, p)$ for $p \in A$. But then by definition of E we get $M \models p$ and $M \models \neg p$, contradiction.

- $r = (a_\xi, a_{\neg\xi})$. But then by definition of E we get $M \models \xi$ and $M \models \neg\xi$, contradiction.
- $r = (a_{\neg\zeta}, a_{\zeta\wedge\xi})$ or $r = (a_{\neg\xi}, a_{\zeta\wedge\xi})$. Then $M \models \zeta\wedge\xi$, and $M \models \neg\zeta$ or $M \models \neg\xi$, contradiction.
- $r = (a_{\zeta\downarrow\xi}, a_{\zeta\vee\xi})$. Then $M \models \zeta \downarrow \xi$, whence $M \models \neg(\zeta\vee\xi)$. But also $M \models \zeta\vee\xi$, contradiction.
- $r = (a_\zeta, a_{\zeta\downarrow\xi})$ or $r = (a_\xi, a_{\zeta\downarrow\xi})$. Then $M \models \zeta \downarrow \xi$, and $M \models \zeta$ or $M \models \xi$. But then also $M \models \zeta \vee \xi$, contradiction.

E attacks all arguments not in E: Let $b \in (A \cup A_\varphi \cup \{a_\perp\}) \setminus E$ be an argument. By definition, there is a formula ψ such that $b = a_\psi$ and $M \not\models \psi$. We use structural induction.

- If $\psi = \perp$ then $a_\varphi \in E$ attacks $a_\perp$ by definition.
- If $\psi = \neg\xi$, then $M \models \xi$ whence $a_\xi \in E$ attacks a_ψ by definition.
- If $\psi = \zeta \wedge \xi$, then $M \models \neg\zeta$ or $M \models \neg\xi$ whence $a_{\neg\zeta} \in E$ or $a_{\neg\xi} \in E$. In any case, E attacks a_ψ by definition.
- If $\psi = \zeta \vee \xi$, then $M \models \zeta \downarrow \xi$ whence $a_{\zeta\downarrow\xi} \in E$ attacks a_ψ by definition.
- If $\psi = \zeta \downarrow \xi$, then $M \models \zeta \vee \xi$ whence $a_\zeta \in E$ or $a_\xi \in E$.

In any case, E attacks a_ψ by definition.

2. Let E be a stable extension of F_φ. Since E is conflict-free, $a_\perp \notin E$. Since E is stable, E attacks $a_\perp$, which yields $a_\varphi \in E$. By Lemma 27, $E \cap A$ is a model of φ. □

In particular, F_φ has no stable extension iff φ is unsatisfiable. While this shows that the construction of Definition 8 works as intended, it remains to show that the number of new arguments is at most linear in the formula size. We can even show that the total increase in size is only linear, thus also the number of new arguments is linear.

Proposition 29. *For any formula φ, we find that $\|F_\varphi\| \in O(\|\varphi\|)$.*

Proof. We first note that

$$\begin{aligned}
\|F_\varphi\| &= \|(A_\varphi \cup A_\perp, R_\varphi \cup \{(a_\varphi, a_\perp)\} \cup R_\perp)\| \\
&= |A_\varphi \cup A_\perp| + |R_\varphi \cup \{(a_\varphi, a_\perp)\} \cup R_\perp| \\
&= |A_\varphi| + 1 + |R_\varphi| + 2 \\
&= |A_\varphi| + |R_\varphi| + 3
\end{aligned}$$

We now use structural induction on φ to show that for all formulas φ, we find $|A_\varphi| \leq 5 \cdot \|\varphi\|$ and $|R_\varphi| \leq 4 \cdot \|\varphi\|$. It then follows that $\|F_\varphi\| \leq (5+4) \cdot \|\varphi\| + 3 = 9 \cdot \|\varphi\| + 3 \in O(\|\varphi\|)$.

$\varphi = \top$:

$$\begin{aligned}
|A_\top| &= |\{a_\top\}| = 1 \leq 5 = 5 \cdot \|\top\| \\
|R_\top| &= |\emptyset| = 0 \leq 4 = 4 \cdot \|\top\|
\end{aligned}$$

$\varphi = \bot$:

$$\begin{aligned} |A_\bot| &= |\{a_\bot\}| = 1 \leq 5 = 5 \cdot \|\bot\| \\ |R_\bot| &= |\{(a_\bot, a_\bot)\}| = 1 \leq 4 = 4 \cdot \|\bot\| \end{aligned}$$

$\varphi = a \in A$:

$$\begin{aligned} |A_a| &= |\{a, a_{\neg a}\}| = 2 \leq 5 = 5 \cdot \|a\| \\ |R_a| &= |\{(a, a_{\neg a}), (a_{\neg a}, a)\}| = 2 \leq 4 = 4 \cdot \|a\| \end{aligned}$$

$\varphi = \neg\xi$:

$$\begin{aligned} |A_\varphi| &= |A_\xi \cup \{a_{\neg\xi}\}| \leq |A_\xi| + 1 \leq (5 \cdot \|\xi\|) + 1 \leq 5 \cdot (\|\xi\| + 1) = 5 \cdot \|\varphi\| \\ |R_\varphi| &= |R_\xi \cup \{(a_\xi, a_{\neg\xi})\}| \leq |R_\xi| + 1 \leq (4 \cdot \|\xi\|) + 1 \leq 4 \cdot (\|\xi\| + 1) = 4 \cdot \|\varphi\| \end{aligned}$$

$\varphi = \zeta \wedge \xi$:

$$\begin{aligned} |A_\varphi| &\leq |A_{\neg\zeta}| + |A_{\neg\xi}| + 3 \leq (|A_\zeta| + 1) + (|A_\xi| + 1) + 3 \\ &\leq (5 \cdot \|\zeta\| + 1) + (5 \cdot \|\xi\| + 1) + 3 = 5 \cdot \|\zeta\| + 5 \cdot \|\xi\| + 5 \\ &= 5 \cdot (\|\zeta\| + \|\xi\| + 1) = 5 \cdot \|\zeta \wedge \xi\| \end{aligned}$$

$$\begin{aligned} |R_\varphi| &\leq |R_{\neg\zeta}| + |R_{\neg\xi}| + 2 \leq (|R_\zeta| + 1) + (|R_\xi| + 1) + 2 \\ &\leq (4 \cdot \|\zeta\| + 1) + (4 \cdot \|\xi\| + 1) + 2 = 4 \cdot \|\zeta\| + 4 \cdot \|\xi\| + 4 \\ &= 4 \cdot (\|\zeta\| + \|\xi\| + 1) = 4 \cdot \|\varphi\| \end{aligned}$$

$\varphi = \zeta \vee \xi$:

$$\begin{aligned} |A_\varphi| &\leq |A_\zeta| + |A_\xi| + 2 \leq 5 \cdot \|\zeta\| + 5 \cdot \|\xi\| + 2 \\ &\leq 5 \cdot \|\zeta\| + 5 \cdot \|\xi\| + 5 = 5 \cdot (\|\zeta\| + \|\xi\| + 1) = 5 \cdot \|\varphi\| \end{aligned}$$

$$\begin{aligned} |R_\varphi| &\leq |R_\zeta| + |R_\xi| + 3 \leq (4 \cdot \|\zeta\|) + (4 \cdot \|\xi\|) + 3 \\ &\leq 4 \cdot \|\zeta\| + 4 \cdot \|\xi\| + 4 = 4 \cdot (\|\zeta\| + \|\xi\| + 1) = 4 \cdot \|\varphi\| \end{aligned}$$ □

Hence under projection, the AF stable extension semantics can realise as much as propositional logic can. With the results of the previous section (AF $\leq_e$ PL), this means that allowing to introduce a linear number of new vocabulary elements (that are later projected out), all languages considered in this paper are equally (universally) expressive.

However, we must note that equal expressiveness does not mean equal efficiency: When we assume that a knowledge base of size n leads to a search space of size $O(2^n)$, then a *linear* increase in knowledge base size (that is, from n to $c \cdot n$ for some constant c) leads to a *polynomial* increase in search space size (that is, from $O(2^n)$ to $O(2^{c \cdot n}) = O((2^n)^c)$.

5. Discussion

We compared the expressiveness of abstract argumentation frameworks, abstract dialectical frameworks, normal logic programs and propositional logic. We showed that expressiveness under different semantics varies for the formalisms and obtained a neat expressiveness hierarchy. These results inform us about the capabilities of these languages to encode sets of two-valued interpretations, and help us decide which languages to use for specific applications. Furthermore, we have seen that the results are sensitive to the vocabulary one is permitted to use, as the hierarchy collapses when we allow to introduce even only a linear number of new atoms.

Concerning succinctness, we have shown that ADFs (under model semantics) are exponentially more succinct than normal logic programs (under supported model semantics), and that even bipolar ADFs (under model semantics) – although being less expressive – can succinctly express some model sets where equivalent normal logic programs (under supported model semantics) over the same vocabulary must necessarily blow up exponentially in size. It is open whether the converse direction also holds, that is, whether BADFs are exponentially more succinct than logic programs (if $\mathrm{LP}^{su} \leq_s \mathrm{BADF}^{su}$) or the two are just mutually incomparable in terms of succinctness (if $\mathrm{LP}^{su} \not\leq_s \mathrm{BADF}^{su}$). For the stable semantics, relative succinctness of logic programs and BADFs is completely open, partly due to the technical aspect that the two stable semantics are conceptually different, as ADFs in fact employ *ultimate* stable models (Denecker et al., 2004; Brewka et al., 2013; Strass & Wallner, 2015). Furthermore, for general ADFs, the computational complexity of the model existence problem of stable semantics is higher than for normal logic programs,[10] so a succinctness comparison with regard to stable models would be of limited significance.

It is easy to see that AFs have a somewhat special role as they are representationally succinct in any case: for a vocabulary A_n, there is syntactically no possibility to specify a knowledge base (an AF) of exponential size, since the largest AF over A_n has size $\|(A_n, A_n \times A_n)\| = n + n^2$ and is thus polynomially large. So anything that can be expressed with an AF can be expressed in reasonable space by definition. However, this "strength" of AFs should be taken with a grain of salt, since they are comparably inexpressive. This can (in addition to the results we presented) already be seen from a simple counting argument: even if all syntactically different AFs over A_n were semantically different (which they are not), they could express at most 2^{n^2} different model sets, which is – for increasing n – negligible in relation to the 2^{2^n} possible model sets over A_n.

In their original paper, Gogic et al. (1995) also used a relaxed version of succinctness, where they allowed to introduce a linear number of new variables. It follows from our results in Section 4 that all formalisms we consider here are equally succinct under this relaxed notion.

Parts of the expressiveness results for normal logic programs carry over to further LP classes. For example, *canonical logic programs* provide a limited form of nesting by allowing literals of the form *not not a* in rule bodies (Lifschitz et al., 1999). This makes it quite easy to see how normal logic programs under supported semantics can be translated to equivalent canonical logic programs, namely by replacing each positive body atom a by *not not a* in

10. Σ_2^P-hard for ADFs (Strass & Wallner, 2015) as opposed to in NP for normal LPs (Bidoit & Froidevaux, 1991; Marek & Truszczyński, 1991).

all rule bodies. Recently, Shen and Zhao (2014) showed that canonical logic programs and propositional logic programs are succinctly incomparable (under an assumption[11]), and also provide interesting avenues for further succinctness studies. We can also add succinctness questions of our own: firstly that of comparing *disjunctive* logic programs under stable models with general ADFs under stable models, since the two have an equally complex (Σ_2^P-complete) model existence problem (Eiter & Gottlob, 1995; Brewka et al., 2013). What is more, there have been alternative proposals for stable model semantics for ADFs:

- ours (Strass, 2013) (Definition 3.2, later called "approximate stable models" by Strass & Wallner, 2015), for which model existence is NP-complete (Strass & Wallner, 2015) and thus potentially easier than that of the stable models of Brewka et al. (2013) (called "ultimate stable models" by Strass & Wallner, 2015);

- the "grounded model" semantics by Bogaerts, Vennekens, and Denecker (2015) (Definition 6.8), whose model existence problem is also Σ_2^P-complete (Bogaerts et al., 2015);

- the "F-stable model" semantics by Alviano and Faber (2015) (Definition 10).

It follows from Theorem 5.9 of Bogaerts et al. (2015) that grounded models and F-stable models coincide. Still, they are demonstrably different from both approximate and ultimate stable models for ADFs (Alviano & Faber, 2015),[12] and their relative succinctness in comparison to normal/disjunctive logic programs is unanalysed.

There is more potential for further work. First of all, a "nice" characterisation of bipolar ADF realisability is still missing; we are unsure whether much improvement over Proposition 11 is possible. Incidentally, for AFs the exact characterisation of compact stable extension realisability constitutes a major open problem (Dunne et al., 2015; Baumann et al., 2014). Second, there are further semantics for abstract dialectical frameworks whose expressiveness could be studied; Dunne et al. (2015) and Dyrkolbotn (2014) already analyse many of them for argumentation frameworks. This work is thus only a start and the same can be done for the remaining semantics. For example the admissible, complete and preferred semantics are all defined for AFs, (B)ADFs and LPs (Strass, 2013; Brewka et al., 2013), and Pührer (2015) has already made a huge step into that direction by characterising realisability. Third, there are further formalisms in abstract argumentation (Brewka et al., 2014) whose expressiveness is by and large unexplored to the best of our knowledge. Finally, the representational succinctness of the subclass of bipolar ADFs (using bipolar propositional formulas to represent them) under supported model semantics is mostly open (cf. Table 2), with some evidence pointing toward meaningful capabilities.

Acknowledgements

This paper combines, extends and improves results of our previous work (Strass, 2014, 2015b, 2015c). We wish to thank Stefan Woltran for providing a useful pointer to related

11. $\mathsf{P} \not\subseteq \mathsf{NC}^1_{/\mathsf{poly}}$, the Boolean circuit equivalent of the assumption $\mathsf{NP} \not\subseteq \mathsf{P}$.
12. In the terminology of Alviano and Faber (2015), approximate stable models (Strass, 2013) are called S-stable models and ultimate stable models (Brewka et al., 2013) are called B-stable models. Both are shown to be different from F-stable models.

work on realisability in logic programming, Bart Bogaerts for pointing out that grounded models and F-stable models are the same, Jörg Pührer for several suggestions for improvement of the manuscript, and Frank Loebe for helpful discussions. This research was partially supported by Deutsche Forschungsgemeinschaft (DFG, project BR 1817/7-1).

References

Al-Abdulkarim, L., Atkinson, K., & Bench-Capon, T. J. M. (2014). Abstract dialectical frameworks for legal reasoning. In Hoekstra, R. (Ed.), *Proceedings of the Twenty-Seventh Annual Conference on Legal Knowledge and Information Systems (JURIX)*, Vol. 271 of *Frontiers in Artificial Intelligence and Applications*, pp. 61–70. IOS Press.

Al-Abdulkarim, L., Atkinson, K., & Bench-Capon, T. J. M. (2015). Evaluating an approach to reasoning with cases using abstract dialectical frameworks. In *Proceedings of the Fifteenth International Conference on Artificial Intelligence and Law (ICAIL)*.

Alviano, M., & Faber, W. (2015). Stable model semantics of abstract dialectical frameworks revisited: A logic programming perspective. In Yang, Q., & Wooldridge, M. (Eds.), *Proceedings of the Twenty-Fourth International Joint Conference on Artificial Intelligence (IJCAI)*, pp. 2684–2690, Buenos Aires, Argentina. IJCAI/AAAI.

Arora, S., & Barak, B. (2009). *Computational Complexity: A Modern Approach*. Cambridge University Press.

Baumann, R., Dvořák, W., Linsbichler, T., Strass, H., & Woltran, S. (2014). Compact argumentation frameworks. In *Proceedings of the Twenty-First European Conference on Artificial Intelligence (ECAI)*, pp. 69–74, Prague, Czech Republic.

Bidoit, N., & Froidevaux, C. (1991). Negation by default and unstratifiable logic programs. *Theoretical Computer Science*, *78*(1), 85–112.

Bogaerts, B., Vennekens, J., & Denecker, M. (2015). Grounded fixpoints and their applications in knowledge representation. *Artificial Intelligence*, *224*, 51–71.

Boppana, R. B. (1986). Threshold functions and bounded depth monotone circuits. *Journal of Computer and System Sciences*, *32*(2), 222–229.

Brewka, G., Dunne, P. E., & Woltran, S. (2011). Relating the semantics of abstract dialectical frameworks and standard AFs. In *Proceedings of the Twenty-Second International Joint Conference on Artificial Intelligence (IJCAI)*, pp. 780–785. IJCAI/AAAI.

Brewka, G., Ellmauthaler, S., Strass, H., Wallner, J. P., & Woltran, S. (2013). Abstract dialectical frameworks revisited. In *Proceedings of the Twenty-Third International Joint Conference on Artificial Intelligence (IJCAI)*, pp. 803–809. IJCAI/AAAI.

Brewka, G., & Gordon, T. F. (2010). Carneades and abstract dialectical frameworks: A reconstruction. In *Proceedings of the Third International Conference on Computational Models of Argument (COMMA)*, Vol. 216 of *FAIA*, pp. 3–12. IOS Press.

Brewka, G., Polberg, S., & Woltran, S. (2014). Generalizations of Dung frameworks and their role in formal argumentation. *IEEE Intelligent Systems*, *29*(1), 30–38. Special Issue on Representation and Reasoning.

Brewka, G., & Woltran, S. (2010). Abstract dialectical frameworks. In *Proceedings of the Twelfth International Conference on the Principles of Knowledge Representation and Reasoning (KR)*, pp. 102–111.

Clark, K. L. (1978). Negation as failure. In Gallaire, H., & Minker, J. (Eds.), *Logic and Data Bases*, pp. 293–322. Plenum Press.

Coste-Marquis, S., Konieczny, S., Mailly, J.-G., & Marquis, P. (2014). On the revision of argumentation systems: Minimal change of arguments statuses. In *Proceedings of the Fourteenth International Conference on Principles of Knowledge Representation and Reasoning (KR)*, pp. 52–61.

Darwiche, A., & Marquis, P. (2002). A knowledge compilation map. *Journal of Artificial Intelligence Research*, *17*, 229–264.

Denecker, M., Marek, V. W., & Truszczyński, M. (2004). Ultimate approximation and its application in nonmonotonic knowledge representation systems. *Information and Computation*, *192*(1), 84–121.

Dimopoulos, Y., Nebel, B., & Toni, F. (2002). On the computational complexity of assumption-based argumentation for default reasoning. *Artificial Intelligence*, *141*(1/2), 57–78.

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*, 321–358.

Dunne, P. E., Dvořák, W., Linsbichler, T., & Woltran, S. (2014). Characteristics of multiple viewpoints in abstract argumentation. In *Proceedings of the Fourteenth International Conference on the Principles of Knowledge Representation and Reasoning (KR)*, pp. 72–81, Vienna, Austria.

Dunne, P. E., Dvořák, W., Linsbichler, T., & Woltran, S. (2015). Characteristics of multiple viewpoints in abstract argumentation. *Artificial Intelligence*, *228*, 153–178.

Dyrkolbotn, S. K. (2014). How to argue for anything: Enforcing arbitrary sets of labellings using AFs. In *Proceedings of the Fourteenth International Conference on the Principles of Knowledge Representation and Reasoning (KR)*, pp. 626–629, Vienna, Austria.

Eiter, T., Fink, M., Pührer, J., Tompits, H., & Woltran, S. (2013). Model-based recasting in answer-set programming. *Journal of Applied Non-Classical Logics*, *23*(1–2), 75–104.

Eiter, T., & Gottlob, G. (1995). On the computational cost of disjunctive logic programming: Propositional case. *Annals of Mathematics and Artificial Intelligence*, *15*(3–4), 289–323.

French, T., van der Hoek, W., Iliev, P., & Kooi, B. (2013). On the succinctness of some modal logics. *Artificial Intelligence*, *197*, 56–85.

Friedman, J. (1986). Constructing $O(n \log n)$ size monotone formulae for the k-th elementary symmetric polynomial of n Boolean variables. *SIAM Journal on Computing*, *15*, 641–654.

Gabbay, D. M. (2011). Dung's argumentation is essentially equivalent to classical propositional logic with the Peirce-Quine dagger. *Logica Universalis*, *5*(2), 255–318.

Gaggl, S. A., & Strass, H. (2014). Decomposing Abstract Dialectical Frameworks. In Parsons, S., Oren, N., & Reed, C. (Eds.), *Proceedings of the Fifth International Conference on Computational Models of Argument (COMMA)*, Vol. 266 of *FAIA*, pp. 281–292. IOS Press.

Gaggl, S. A., Rudolph, S., & Strass, H. (2015). On the computational complexity of naive-based semantics for abstract dialectical frameworks. In Yang, Q., & Wooldridge, M. (Eds.), *Proceedings of the Twenty-Fourth International Joint Conference on Artificial Intelligence (IJCAI)*, pp. 2985–2991, Buenos Aires, Argentina. IJCAI/AAAI.

Gebser, M., Kaminski, R., Kaufmann, B., Ostrowski, M., Schaub, T., & Schneider, M. (2011). Potassco: The Potsdam Answer Set Solving Collection. *AI Communications*, *24*(2), 105–124. Available at `http://potassco.sourceforge.net`.

Gelfond, M., & Lifschitz, V. (1988). The stable model semantics for logic programming. In *Proceedings of the International Conference on Logic Programming (ICLP)*, pp. 1070–1080. The MIT Press.

Gogic, G., Kautz, H., Papadimitriou, C., & Selman, B. (1995). The comparative linguistics of knowledge representation. In *Proceedings of the Fourteenth International Joint Conference on Artificial Intelligence (IJCAI)*, pp. 862–869. Morgan Kaufmann.

Jukna, S. (2012). *Boolean Function Complexity: Advances and Frontiers*, Vol. 27 of *Algorithms and Combinatorics*. Springer.

Lifschitz, V., & Razborov, A. (2006). Why are there so many loop formulas?. *ACM Transactions on Computational Logic*, *7*(2), 261–268.

Lifschitz, V., Tang, L. R., & Turner, H. (1999). Nested expressions in logic programs. *Annals of Mathematics and Artificial Intelligence*, *25*(3–4), 369–389.

Lin, F., & Zhao, Y. (2004). ASSAT: Computing answer sets of a logic program by SAT solvers. *Artificial Intelligence*, *157*(1-2), 115–137.

Linsbichler, T. (2014). Splitting abstract dialectical frameworks. In Parsons, S., Oren, N., & Reed, C. (Eds.), *Proceedings of the Fifth International Conference on Computational Models of Argument (COMMA)*, Vol. 266 of *FAIA*, pp. 357–368. IOS Press.

Marek, V. W., & Truszczyński, M. (1991). Autoepistemic logic. *Journal of the ACM*, *38*(3), 587–618.

Osorio, M., Zepeda, C., Nieves, J. C., & Cortés, U. (2005). Inferring acceptable arguments with answer set programming. In *Proceedings of the Sixth Mexican International Conference on Computer Science (ENC)*, pp. 198–205.

Polberg, S. (2014). Extension-based semantics of abstract dialectical frameworks. In Endriss, U., & Leite, J. (Eds.), *Proceedings of the Seventh European Starting AI Researcher Symposium (STAIRS)*, Vol. 264 of *FAIA*, pp. 240–249. IOS Press.

Polberg, S., Wallner, J. P., & Woltran, S. (2013). Admissibility in the abstract dialectical framework. In Leite, J., Son, T. C., Torroni, P., van der Torre, L., & Woltran, S. (Eds.), *Proceedings of the Fourteenth International Workshop on Computational Logic in Multi-Agent Systems (CLIMA XIV)*, Vol. 8143 of *LNAI*, pp. 102–118. Springer.

Pührer, J. (2015). Realizability of three-valued semantics for abstract dialectical frameworks. In Yang, Q., & Wooldridge, M. (Eds.), *Proceedings of the Twenty-Fourth International Joint Conference on Artificial Intelligence (IJCAI)*, pp. 3171–3177. IJCAI/AAAI, Buenos Aires, Argentina.

Shen, Y., & Zhao, X. (2014). Canonical logic programs are succinctly incomparable with propositional formulas. In *Proceedings of the Fourteenth International Conference on the Principles of Knowledge Representation and Reasoning (KR)*, pp. 665–668, Vienna, Austria.

Strass, H. (2013). Approximating operators and semantics for abstract dialectical frameworks. *Artificial Intelligence*, *205*, 39–70.

Strass, H. (2014). On the relative expressiveness of argumentation frameworks, normal logic programs and abstract dialectical frameworks. In Konieczny, S., & Tompits, H. (Eds.), *Proceedings of the Fifteenth International Workshop on Non-Monotonic Reasoning (NMR)*.

Strass, H. (2015a). Instantiating rule-based defeasible theories in abstract dialectical frameworks and beyond. *Journal of Logic and Computation*, Advance Access published 11 February 2015, http://dx.doi.org/10.1093/logcom/exv004.

Strass, H. (2015b). The relative expressiveness of abstract argumentation and logic programming. In *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence (AAAI)*, pp. 1625–1631, Austin, TX, USA.

Strass, H. (2015c). Representational succinctness of abstract dialectical frameworks. In Black, E., Modgil, S., & Oren, N. (Eds.), *Proceedings of the Third International Workshop on Theory and Applications of Formal Argumentation (TAFA)*.

Strass, H., & Wallner, J. P. (2015). Analyzing the computational complexity of abstract dialectical frameworks via approximation fixpoint theory. *Artificial Intelligence*, *226*, 34–74.

Tseitin, G. S. (1968). On the complexity of derivations in the propositional calculus. *Structures in Constructive Mathematics and Mathematical Logic, Part II, Seminars in Mathematics (translated from Russian)*, 115–125.